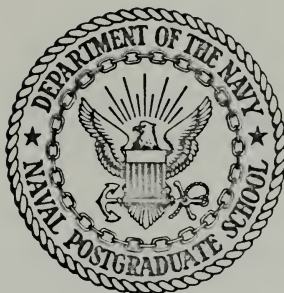


CONVERSION OF A TWO-DIMENSIONAL MAGNETO-
HYDRODYNAMIC COMPUTER CODE FROM CDC-
6400 FORTRAN TO IBM 360/67 FORTRAN

Charles Stanley Yager

NAVAL POSTGRADUATE SCHOOL

Monterey, California



THESIS

CONVERSION OF A TWO-DIMENSIONAL
MAGNETOHYDRODYNAMIC COMPUTER CODE FROM
CDC-6400 FORTRAN TO IBM 360/67 FORTRAN

by

CHARLES STANLEY YAGER JR

Thesis Advisor:

G. A. Garrettson

June 1972

Approved for public release; distribution unlimited.

Conversion of a Two-Dimensional
Magnetohydrodynamic Computer Code From
CDC-6400 FORTRAN to IBM 360/67 FORTRAN

by

Charles Stanley Yager Jr.
Captain, United States Army
B.S., Rose Polytechnic Institute, 1964

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN PHYSICS

from the

NAVAL POSTGRADUATE SCHOOL
June 1972

ABSTRACT

In his Ph.D. thesis Doctor Irvin R. Lindemuth, at the University of California/Livermore, developed a very general method for numerically solving two-dimensional, two-fluid magnetohydrodynamic equations. A copy of his computer code was given to the Physics Department at the Naval Postgraduate School for conversion to the IBM 360/67 system presently in operation at the school. This paper is intended to be a users manual for this code. Numerous changes to the original code were required due to the inherent differences between the CDC and IBM machines. The conversion of this code as well as a complete understanding of its operation and logic was the goal in the preparation of this paper.

TABLE OF CONTENTS

I.	Introduction	6
	A. Background	6
	B. Purpose	7
	C. Organization of the Paper	8
II.	Conversion Procedure	8
	A. Plan	8
	B. Problems Encountered During Compilation ...	9
	1. Programming Language Differences	9
	2. Procedure Limitations	9
	C. Problems Encountered During Execution	10
	1. Core Limitation	10
	2. Subroutine Calls	10
	3. Data Sets	11
	4. Abnormal Ends (ABEND)	11
	5. Typographical Errors	12
III.	Program Organization	13
	A. Original Code	13
	B. Present Code	15
IV.	Use of the CODE	15
	A. Present Use	15
	B. Future Use	17
V.	Test Runs and Results	18
VI.	Error Messages and Diagnostics	19

VII.	Conclusions	20
A.	Program Conversions	20
B.	Ideas to Pursue	21
1.	Storage of Code on Disc	21
2.	Time Sharing	21
3.	Reduction of the Source Code	22
4.	Data Retention	23
APPENDIX A:	Machine Differences	24
APPENDIX B:	Job Control Language (JCL) Cards	34
APPENDIX C:	Files	43
APPENDIX D:	Subroutine Descriptions	48
APPENDIX E:	Input Data Card Formats	67
APPENDIX F:	COMMON Block Storage and Equivalence ...	74
APPENDIX G:	Array Descriptions	83
APPENDIX H:	Data Sets	91
APPENDIX I:	Identifier Descriptions	100
CONVERTED COMPUTER PROGRAM	105
LIST OF REFERENCES	211
INITIAL DISTRIBUTION LIST	212

LIST OF TABLES

I.	Identifier Name Changes	25
II.	Data Set Identification Reference	47
III.	COMMON Block Construction	79
IV.	Equivalence Reference of Important Variables	82

I. INTRODUCTION

A. BACKGROUND

In his Ph.D. thesis, THE ALTERNATING-DIRECTION IMPLICIT NUMERICAL SOLUTION OF TIME-DEPENDENT, TWO-DIMENSIONAL, TWO-FLUID MAGNETOHYDRODYNAMIC EQUATIONS, Doctor Irvin R. Lindemuth developed a very general and complex computer code which he made available to the physics department at the Naval Postgraduate School. The code was originally written in CDC FORTRAN for use on the CDC 3400 computer. This machine has a relatively small magnetic core which makes available to the user only about 28,000 words of memory. Procedure modifications required to conserve core storage and the generality of the code contributed to its complexity. With the proper selection of input parameters the code can solve one or two dimensional problems in either rectangular, cylindrical, or spherical coordinates. It also provides extensive control of data transfer between the CPU and its associated I/O devices. Diagnostic subroutines are available to assist in detecting numerical instabilities and inaccuracies during execution of the code. An additional feature is the restart capability which allows the program to pick up execution at the point where it was terminated or at any previous time step which was stored in external memory.

The code was obtained in the form of a deck of punched cards. This deck consisted of approximately 5500 source

cards. A test program, sample output, Lindemuth's thesis, and a brief description of the code were obtained at the same time. The author and his advisor, Doctor G. A. Garrettson, consulted with Doctor Lindemuth on several occasions to discuss the complexities and subtleties of the code. With these sources of data and information, the conversion process was begun.

This code is of particular interest to the Physics Department at the Naval Postgraduate School where projects researching laser produced plasmas confined in a magnetic field and theta-pinch plasma devices are currently under way. It has verified the shell structure of a laser-produced plasma observed by Schwirzke and should be a valuable tool in future research. For a detailed explanation of the equations and numerical methods incorporated into this code, the reader is referred to Lindemuth's thesis (UCRL 51103).

B. PURPOSE

This thesis work was strictly concerned with converting Lindemuth's code to a form which could be used on the school's IBM 360/67 computer. As a result this paper is intended to be a users manual for the code and will be devoted primarily to the construction of the code along with detailed descriptions of subroutines, files, arrays, variables, input parameters, and so forth.

C. ORGANIZATION OF THE PAPER

Consistant with the subject, this paper is organized like a computer code which has a small main program and many subroutines. The main body of the thesis consists of short sections sketching the plan of attack for the conversion, a few comments on problems encountered, operation and use of the code, and a few conclusions regarding the conversion. The bulk of the presentation is contained in the appendices. These appendices contain detailed descriptions of particular elements such as arrays, subroutines, common blocks, input parameters, and so forth.

II. CONVERSION PROCEDURE

A. PLAN

Since the code was written in FORTRAN programming language, it was expected to be reasonably compatable with the IBM machine. The plan for conversion was to place the source code in the IBM FORTRAN compiler and see what syntax differences existed. These differences would be resolved and the source code resubmitted to the compiler. This process would continue until the entire program could be compiled without error diagnostics. At this point the program would be loaded and execution would be attempted with the test problem input furnished by Lindemuth.

B. PROBLEMS ENCOUNTERED DURING COMPILATION

After the first compilation attempt, it became apparent that the conversion would be considerably more complicated and time consuming than anticipated. The difficulties divided themselves into two categories. One was differences in the FORTRAN programming language between the CDC and IBM machines and the other was limitations in the IBM catalogued procedures for compilation.

1. Programming Language Differences

The variations in programming languages were apparent in the source listing output of the compiler. A detailed list of the language differences and the action taken to resolve them is presented in appendix A.

2. Procedure Limitations

Most of the time spent in obtaining a syntactically correct source deck was due to turn-around time and non-productive runs caused by unforeseen limitations in the catalogued procedures used to compile the source deck. Appendix B discusses Job Control Language (JCL) cards and catalogued procedures in detail. The majority of the problems were the direct result of the enormous size of the code. Lindemuth's source code contained close to 6000 cards after the multiple command cards were separated into single cards. Working with individual subroutines eliminated most of the size limitation problems during compilation.

Another size limitation was experienced during the compilation of subroutine MAT2. This subroutine was so large it exceeded the CPU core allocated by the catalogued procedure. An over-ride statement allocated additional core and the compilation then continued to completion.

C. PROBLEMS ENCOUNTERED DURING EXECUTION

After the code was successfully compiled an execution attempt was made. At that point several additional problems arose.

1. Core Limitation

At first the program failed to load into core because the catalogued procedure for execution limits the program and all external references to 100K bytes of core memory. A check of the linkage editor output showed that the code required over 200K bytes of memory to load. This resulted in a non-productive run, but was easily corrected by over-riding the catalogued procedure's REGION declaration.

2. Subroutine Calls

It was discovered during a subsequent execution attempt that call were being made to subroutines which could not be located by the linkage editor. It was at this point that an additional discrepancy in programming languages was discovered. CDC FORTRAN uses different identifiers for the subroutines in the standard Scientific Subroutine Package (SSP) which forms a portion of the software accompanying the machine. For example, IBM FORTRAN

would call EXP(arg), while CDC FORTRAN would call EXPF(arg). After all of the subroutine identifiers had been corrected the code compiled and link edited properly with a REGION requirement of 214K bytes.

3. Data Sets

Lindemuth's code originally provided for creation and destruction of files, or external data sets, by calling a utility subroutine CREATE which is available in CDC FORTRAN. Appendix C discusses files and data sets in detail. The IBM system requires that the data sets be declared and defined before execution of the program. Therefore the number, size, and description of data sets had to be determined and entered before the program could be executed. All coding which involved the CREATE subroutine was removed from the code.

4. Abnormal Ends (ABEND)

During another execution attempt an "underflow" ABEND was received in subroutine TCINIT. This again was the result of machine differences. The CDC machine can handle floating point numbers in the range 10^{+322} to 10^{-293} while the IBM machine will only handle number to the order of 10^{+75} . In this subroutine a constant was being evaluated which involved the multiplication of several very small numbers followed by division of some very small numbers. Although the final result was well within the IBM limit, at certain points of the calculation the accumulation register

was required to store a number whose value was outside the limit and the ABEND resulted. The IBM control program has a STANDARD FIXUP subroutine which assigns a precision zero to the register which underflows and then continues execution of the program. This, of course, is unacceptable in these calculations and frequently leads to another ABEND when division by this constant is attempted. This problem was solved by rewriting the FORTRAN statement in such a way that the accumulation register never exceeds the limits during the calculation.

5. Typographical Errors

This is perhaps the most elusive error encountered. A tremendous amount of rewriting was done to the original code. During these revisions a number of typographical errors occurred. Some of these were discovered while studying a section of the code on a subject completely unrelated to the error. This type of discovery is fortunate but largely a matter of luck. These errors have usually been discovered while looking for the cause of an ABEND. For example, during an execution attempt the program terminated with a completion code OC4 which indicates that there is a subscripting error where the program was ABENDED. However this message by itself gives no indication of where in the program the execution terminated. The program was run again with a SYSUDUMP included in the JCL cards. Upon an ABEND the contents of core and all of the registers are

printed out. Among many other things the SYSUDUMP output contains a program source work (PSW) which gives the location of the command which caused the ABEND. The PSW and the linkage editor map can then locate the troublesome subroutine. Examination of the subroutine revealed the error in an EQUIVALENCE statement where NADDA, an index for a subscripted variable, had been typed MADDA. Since MADDA had not been initialized in the subroutine, the computer picked whatever value was in its storage location and attempted to use it as the index. It was highly improbable that this value was an integer and even less probable that it was within the range of the array dimension. Therefore the program naturally ABENDED.

III. PROGRAM ORGANIZATION

A. ORIGINAL CODE

The original code consists of the main program and thirty-three subroutines. These subroutines can be placed in the following categories:

- Initialization
- Logic and Control
- Operation
- Output
- Diagnostic

These categories are discussed in appendix D which also contains a detailed explanation of each subroutine.

Execution options in the program are determined by parameters entered on input cards. Appendix E contains

a list of all the input cards; this list describes the card format, calling routine and statement number, and variable identification.

Data is passed between subroutines by the use of COMMON blocks of storage which are then equivalenced to operating variables in the subroutines. Appendix F contains a list and description of the COMMON blocks used in this program.

The computed values of the dependent variables are stored in the VAR vector of COMMON/C3/. This vector together with the DADT vector, which contains time step information, is buffered out to external permanent storage after each time step. The DA vector and the RZ vector which contain startup information and mesh point location, respectively, are buffered out initially to provide a restart capability at any past time step in the program execution. Appendix G contains a list and detailed description of all the important vectors and arrays used by this program.

The CDC version of this code provides for some external control during program execution by using "sense switches." These are actual switches located on the operator's console that flag a preset location which can be tested by a logical GO TO command. Thus some degree of control can be exercised during execution of the program. These switches are used in conjunction with subroutines CHECK and SPECHK to get convenient diagnostics for checking numerical instabilities.

The code is set up to buffer out data to storage after each time step and to print the data after each nth time step as determined by the input parameter NDTPNT.

B. PRESENT CODE

The present code consists of the main program and thirty-four subroutines. The organization is essentially the same as the original except that all sense switch capability has been removed. Use of the diagnostic subroutines must be determined before program execution and requests entered on the input cards. A modification has been added to allow the code to buffer out data to storage after each mth time step as determined by the input parameter NDTBUP. The additional subroutine was added to handle binary data transfer in and out of external storage. The converted code is listed beginning on page 105.

IV. USE OF THE CODE

A. PRESENT USE

During the testing and initial run phases the code has been run using punched cards as input to OS. The source code was compiled to produce an object deck which was then used to input the program to the system 360 computer. Using the object deck as input has the advantages of reading in less cards to start the program and requiring less set-up time (compilation and link editing of source code) before execution. For example the compilation step takes

approximately three and one-half minutes of CPU time while the link step takes only four seconds.

The test program was run using a JCL deck of the form

```
//YAG0833 JOB (0833,0739FP,FA11),'THESIS'  
// EXEC FORTLG  
//LINK.SYSIN DD *  
  (set of object decks)  
/*  
//GO.FT06F001 DD SPACE=(CYL,(3,1))  
//GO.FT10F001 DD UNIT=2314,DSNAME=&RECORD,          C  
  DCB=(RECFM=FB,LRECL=100,BLKSIZE=3500),          C  
  VOL=SER=MARY,SPACE=(TRK,(15,1),RLSE),          C  
  DISP=(NEW,DELETE)  
//GO.SYSIN DD *  
  (test program input deck)  
/*
```

The JCL program format is discussed in appendix B. The data set definition card which begins //GO.FT10F001 and the data set over-ride card which begins //GO.FT06F001 are discussed in detail in appendix H.

All program runs will use this same general format. Program modification will be performed via the input deck. Appendix E lists all of the input cards and describes them completely. All of the input cards listed must be included in the input stream in the listed order, even though the parameters may all be zero. Two exceptions to this rule are also discussed in appendix E.

To determine the input parameters the problem must be formulated and initial and boundary conditions fixed. The appropriate section of reference 1 (for example, application to laser produced plasmas, application to theta pinch) will

assist the operator in determining what input parameters are required. After the problem is formulated, use of appendix E in conjunction with appendix D (subroutine descriptions) will determine which input cards and parameters to include in the input data set stream.

The output format is composed of two parts - the input data and the computed values of the dependent variables. The input data listing is a labeled listing of all the variables, boundary conditions, and initial conditions which are read in the input data set. The computed values are separated into sections by dependent variable. Each section contains a plot of the dependent variable in the Z plane following by a matrix listing of the value of the dependent variable at each mesh point. This last section is printed each time OUTPT1 is called as determined by NDTFNT.

B. FUTURE USE

Once the code has been completely tested and is operating satisfactorily it will be placed on a permanent external storage device. Methods for this storage are discussed in reference 8. This will have the advantages of less handling of card decks and increased access time and flexibility. When the code is placed on external storage, it can be stored in a form which can be loaded directly into core without compilation or link editing. Thus by using only a few cards in the JCL program, the entire program code can be entered and made ready for execution in less than a

second. The only additional cards to be manually entered would be the input data cards which would be determined as before.

V. TEST RUNS AND RESULTS

At the time of this writing the test program had only partially executed successfully. The converted code did successfully execute the initializing functions and sub-routines which set up initial conditions for all of the variables at each mesh point and established proper boundary conditions. The execution ran into difficulty when data transfer into external storage was attempted. It is anticipated that this problem can be quickly solved and the program executed through normal completion.

A printout of the test program output was furnished with the original code. A copy of this printout is included as part of reference 9 which has been given to Doctor G. A. Garrettson of the Physics Department at the Naval Postgraduate School. In addition to the test results, this printout contains notes explaining the contributions of specific subroutines to the output format. Reference 9 also contains a listing of the original code, the initial test results of the converted program, and a compilation listing of the converted program which includes an object code listing for debugging operations.

VI. ERROR MESSAGES AND DIAGNOSTICS

A complete treatment of error messages and diagnostics is beyond the scope of this paper. Reference 10 contains detailed discussions of all the error messages and completion codes, their meanings, probable causes for the message, and possible corrective actions.

Reference 9 contains notes on the use of object code listings and SYSDUMP (a core dump upon an ABEND) printouts to locate a statement which is causing an error. To get a SYSDUMP insert the JCL card `//GO.SYSDUMP DD SYSOUT=A` immediately before the `//GO.SYSIN DD *` card in any of the execution programs. The single most important piece of information to be extracted from the dump is the program source word (PSW). The last six characters of this word give the location address (in hexadecimal) in core of the command which caused the ABEND. Subtracting the entry address of the program from this ABEND address will give the location in the program of the bad command. This reference address can be used with the control section map in the linkage editor output to locate the subroutine which contains the bad command. To locate the exact statement which is causing the error subtract the origin of the subroutine from the reference address. This will give the address (in hexadecimal) of the bad command in its subroutine. Now by locating this address in the object code listing, the statement number of the bad command can be extracted.

The computer center provides duty programmers to assist the operator in interpreting error messages which he cannot understand. They will explain generally what is causing the error and refer the operator to a specific reference which deals with the problem. If an error cannot readily be located, much time will be saved by consulting these programmers immediately.

VII. CONCLUSIONS

A. PROGRAM CONVERSIONS

Conversion of computer codes can be an extremely complex and time consuming task. Often it is a difficult task to convert a code which is to be used on the same type of machine in different installations. When the machines are of different manufacture the complications can reach enormous proportions, even though the code is written in the same programming language. It is frequently more advantageous to write a new code than to convert the old one.

A good flow chart for a code is an invaluable tool. Without it, the programmer may spend excessive time trying to understand an operation which would be obvious if he had a flow chart which showed the logic of the operation. Plenty of comment cards are beneficial, especially if the code is to be used by others.

The decision to convert a computer code should be made carefully. The fact that it runs on another machine should not be the primary basis for the decision. Consideration must be given the machines, their I/O devices, the respective programming languages, the amount of change that will be required in the code, the storage areas available, and the speed of execution. If the code is well documented, it may well be better to rewrite the code and avoid the pitfalls which inevitably accompany conversions of large complex codes.

B. IDEAS TO PURSUE

1. Storage of Code on Disc

The most efficient method of entering the code into the CPU is by creating a load module library which is discussed in reference 8. A load module is the output of the linkage editor. For execution, a call is made to this library and the code is immediately placed in core and is ready for execution. The only additional cards to be manually entered will be the input data cards.

2. Time Sharing

A full discussion of time sharing is beyond the scope of this paper. However, its possible use should be mentioned at this point. Time sharing is available during specific periods of the day and a user may sign up for 30 minutes at a time. During time sharing the operator may communicate directly with the machine. This would allow reinsertion of the manual control features of the code which

had to be removed for operation on OS.

Space requirements would be the limiting consideration. In its present form the code is too large to fit in the files allocated to a particular terminal. It is possible that a special authorization and allocation could be obtained to allow operation on the time sharing system.

Time sharing operation would allow the operator to monitor critical variables during the execution of the code and to observe any instabilities as they develop. This would allow early diagnosis of the problems associated with the instability and save much computing time that would be wasted on the OS system.

3. Reduction of the Source Code

Lindemuth's source code was written towards the goal of successful execution and had not been fully optimized when received at the Naval Postgraduate School. If time and space become critical, many of the procedures can be rewritten in shorter form or in a time-saving form. This would be a time-consuming task initially, but would provide great savings in core storage and execution time in subsequent runs.

Another space-saving recommendation is to specialize the program. In its present form the code is written to solve many general problems. After the code is fully debugged and operating properly, it would be advantageous to rewrite the code for a specific problem, for instance the laser produced plasma problem. This would allow all

of the coding which was not used for the specific problem to be removed and thus reduce the size of the storage area required for execution.

4. Data Retention

Presently the code stores the values of all the dependent variables at each mesh point for every mth time step as specified on input card number 3. This can quickly consume large areas of storage. If retention of this data is required for future data analysis, consideration should be given to storing the information on larger and less frequently used external storage devices. Magnetic tapes would be good for this purpose since the user could obtain his own personal tapes and use them in any way he required. Data could be placed on these tapes by either of two methods.

If long runs are expected (for example runs of from one to two hours during the night or on week-ends) the code could be modified to write file RECORD on the tape instead of on the disc as is presently done.

If short runs are expected (especially if time sharing can be implemented) it would be advantageous to write on the disc for several sessions and then use a utility routine to transfer the data to a tape. This is because the access time to the disc is much shorter than to the tape and this time could be used for execution during a short session.

APPENDIX A

MACHINE DIFFERENCES

This program was originally written in CDC FORTRAN for use on the CDC-3400 and later modified to run on the CDC-6400. While FORTRAN is generally regarded as a universal programming language, there are a great many differences in the versions written for machines of different manufacture. The following list compares major differences between the CDC 3400/6400 and the IBM 360 FORTRAN versions which were discovered during the conversion work on Lindemuth's code. Corrective action taken for each item follows the description.

1. Identifiers

- a) CDC allows up to eight alphanumeric characters.
- b) IBM allows up to six alphanumeric characters. This required locating all identifiers over six characters in length, renaming them with six character strings, and entering these changes in the source deck. Table I lists all of the oversize identifiers and their new names.

2. COMMON Blocks

- a) CDC allows alphanumeric identifier names up to eight characters in length, or numeric identifiers up to eight characters.
- b) IBM allows only alphanumeric identifiers up to six characters in length, the first character being

TABLE I
IDENTIFIER NAME CHANGES

ORIGINAL		MODIFIED	ORIGINAL		MODIFIED
BCRASHK	..	BCRSHK	DBPEPDZ	..	BPEPDZ
BCZASHK	..	BCZSHK	DBREPDR	..	BREPDR
BEXINIT	..	BEXINT	DBREPDZ	..	BREPDZ
BFROMSI	..	BFRMSI	DBZEPDR	..	BZEPDR
BOUNDRY	..	BOUNDY	DBZEPDZ	..	BZEPDZ
BZEINIT	..	BZEINT	DSIEPDR	..	SIEPDR
CBERMZM	..	BERMZM	DSIEPDZ	..	SIEPDZ
CBERPZM	..	BERPZM	DTCDVCX	..	DTDVDX
CBERMZP	..	BERMZP	GAM10U2	..	GM1U02
CBERPZP	..	BERPZP	IBEDATA	..	IBEDAT
CLNLM2	..	CLNLM2	INITCON	..	INTCON
CTCRMZM	..	TCRMZM	IPLAVAC	..	IPLAVC
CTCRMZP	..	TCRMZP	ISTEPNO	..	ISTPNO
CTDRMZM	..	TDRMZM	ITCDATA	..	ITCDAT
CTDRMZP	..	TDRMZP	ITCVERS	..	ITCVER
CTDRPZM	..	TDRPZM	IOUT1MW	..	IOT1MW
CTDRPZP	..	TDRPZP	JSTEPNO	..	JSTPNO
D2BPDZ2	..	D2BPDZ	LNLAM2E	..	LNLAM2E
D2BRDR2	..	D2BRDR	LNLAM2I	..	LNLAM2I
D2BZDR2	..	D2BZDR	NOTCDRV	..	NTCDRV
D2RODR2	..	D2RODR	OUTPUT1	..	OUTPT1
D2SIDR2	..	D2SIDR	OVERISU	..	OVRISU
D2TEDR2	..	D2TEDR	PRNCHR	..	PRNCHR
D2TIDR2	..	D2TIDR	RORINIT	..	RORINT
D2VPDR2	..	D2VPDR	RUNDATA	..	RUNDAT
D2VRDR2	..	D2VRDR	SETKEQO	..	STKEQO
D2VZDR2	..	D2VZDR	STARTUP	..	STRTUP
DBPEPDR	..	BPEPDR	STARTYM	..	STRTYM

alphabetic. Lindemuth's code used numbered COMMON blocks (COMMON/1/). This required changing all COMMON block identifiers to an alphanumeric identifier. This was accomplished by prefixing each numeric identifier with a 'C' (COMMON/C1/).

3. Multiple Commands

a) CDC allows more than one command to be placed on one card by placing a '\$' between the commands. For example: A=1 \$ B=2 \$ C=3

b) IBM allows only one command per card. Due to the large number of multiple command cards, a short program was written to locate these cards, separate the commands, place them on separate cards, and renumber the source deck.

4. Computed GO TO

a) CDC version takes the form GO TO (a,b,c) X.

b) IBM version takes the form GO TO (a,b,c), X. This required locating all of the computed GO TO commands and inserting a comma after the right parenthesis.

5. Floating Point Constants

a) CDC allows floating point constants to range from 10^{+322} to 10^{-293} .

b) IBM allows floating point constants in the range 10^{+75} . While the results of calculations are not outside the range, intermediate values of some calculations do

fall beyond this range resulting in UNDERFLOW or OVERFLOW and subsequent termination of program execution. This was corrected by locating such calculations during test runs and rewriting the functions such that overflow or underflow would not occur.

6. I/O Commands

a) CDC I/O commands take the form READ/WRITE INPUT TAPE a,b,list where 'a' is the data set identifier, 'b' is the format statement number, and 'list' is the set of variables to be read or written.

b) IBM I/O commands take the form READ/WRITE(a,b) list where 'a', 'b', and 'list' are defined above. This required locating all READ/WRITE INPUT TAPE commands and rewriting them in the IBM format. It was not practical to write a program to accomplish this because the CDC I/O command did not always appear in the same location on the card.

7. Transferring Bulk Data

a) CDC FORTRAN has a data transfer command of the form BUFFER IN/OUT (a,n)(A,B) where 'a' is the device number, 'n' is the direction mode, 'A' is the first word to be read or written, and 'B' is the last word to be read or written. This command causes all of the words between 'A' and 'B' to be transferred into or out of core in the binary mode. The parameter 'n' will specify whether data is transferred beginning with 'A' and ending with 'B' or vice versa.

b) IBM FORTRAN provides no such command. To transfer binary data in or out of core is accomplished by a command of the form READ/WRITE(n) list where 'a' is the device number and 'list' is the list of words to be transferred. Each item must be either listed explicitly or implicitly within an implied DO loop. Conversion required locating each BUFFER IN/OUT command and changing it to the IBM READ/WRITE(a) command. Again it was not feasible to write a program for machine conversion.

8. I/O Buffering Check

a) CDC checks the status of a BUFFER IN/OUT operation with a command of the form IF(UNIT,a) n,m where 'a' is the device number and 'n' and 'm' are statement numbers. This command looks at unit 'a' which had previously been given a BUFFER command. If the unit is still transferring data, program control is transferred to statement number 'n'. If the buffering is complete, program control is transferred to statement number 'm'. This allows the CPU to work on another section of the program which does not involve the data being transferred. In this way, the CPU is not tied up waiting for the slow I/O function to be completed.

b) IBM does not have this command. Conversion consisted of simply deleting these commands from the source deck.

9. Files

a) CDC FORTRAN contains a subroutine which will create, destroy, open, or close a file during execution of the program. Input parameters for this subroutine may be variable identifiers.

b) IBM FORTRAN has no such subroutine. File size and description must be determined by the operator before the program is compiled. File declarations must be entered on the JCL cards. Conversion consisted of deleting all calls to this subroutine from the source deck. In addition, special instructions for creation of files and data sets were written.

10. Word Length

a) CDC machines use 48 bit words.

b) IBM machines use 32 bit words. This has indirect effects which are included in other subjects below.

11. Hollerith Assignment

a) CDC FORTRAN allows assignment statements of the form `IWORD=8Hword` where 'word' is a Hollerith string of eight characters.

b) IBM FORTRAN does not allow direct Hollerith assignment. Such assignments must be made with a DATA statement. Since many of the assignments were to dimensioned variables, the following conversion was used. All Hollerith strings were assigned to identifiers in a DATA statement. The identifiers were of the form `COMT1`,

COMT2, and so forth. In the program the subscripted variables were assigned the new identifiers instead of the Hollerith strings. An example is: IDA(1)=COMT1 where COMT1='word' instead of IDA(1)=8Hword.

12. Hollerith Word Size

a) Because of the 48 bit word and BCD format, CDC can assign an eight character Hollerith string to a REAL*4 integer variable.

b) Because of the 32 bit word and EBCDIC format, IBM can only assign a four character Hollerith string to a REAL*4 variable. A two step conversion was required. First, all variables containing Hollerith strings were declared REAL*8. This satisfied assignment requirements. A second problem arose in COMMON/C3/. The IDA array is equivalenced to DA(71) in this common block. The DA array is type REAL*4 and the IDA array had been changed to type REAL*8. Therefore the DA array had to be extended by forty words to accommodate the double precision IDA array. In addition all variables which were equivalenced to DA(111) or higher had to be equivalenced forty words higher to avoid interference with the expanded IDA array.

13. Sense Switches

a) CDC 3400 provides a capability of entering a '0' or '1' into specified locations during execution of the program without a PAUSE or program stop. This is

accomplished by manually positioning switches at the operators console. The command IF SENSE SWITCH (n) m will cause program control to be transferred to statement number 'm' if sense switch 'n' is on. Otherwise control will come from the next executable command.

b) No sense switch capability as described above is available to the general user on the IBM operating system. Conversion consisted of removing all sense switch commands from the source program, along with much of the diagnostic and manual control capability. Presently the operator must know what diagnostics or special control features he wants and enter these requests on input data cards before he runs the program.

14. Pause

a) CDC allows use of the PAUSE command which will cause the program to halt execution and await instruction from the operator's console. This command was used extensively with the diagnostic subroutines and manual control options mentioned in paragraph 13 above.

b) IBM 360 has the PAUSE capability, but does not permit its use by users on the operating system (OS). If the code can be made compatible with the CP/CMS system (time sharing), the PAUSE command can be used to advantage as intended.

15. Character Representation

a) Lindemuth's source code was punched in BCD.

b) Although the IBM machine can convert from BCD to EBCDIC for computation, all of the periferal equipment uses EBCDIC representation. Conversion of the source deck was made by the same program discussed in paragraph 3 above.

16. Comment Card Lists

a) CDC allows an unlimited number of consecutive comment cards to be placed in a source program deck.

b) IBM allows only 31 consecutive comment cards to be placed in a source program deck. If more than 31 comment cards are in a program deck during compilation, 31 cards will be listed, followed by an error message terminating the listing of the comment cards. Compilation will proceed with the next executable statement. Lists longer than 31 cards may be printed by inserting an executable statement every 31 cards. The command CONTINUE is sufficient for this purpose.

17. Format Statements

a) CDC FORTRAN limits a Hollerith string within a format statement by inclosing the string with asterisks.

b) IBM limits the string with single quotes. Conversion required changing all of the limiting asterisks to single quotes.

18. Integer Size

a) CDC's 48 bit word allows integers up to $2^{59}-1$.

b) IBM's 32 bit word allows integers up to $2^{31}-1$.

This caused integer constants used to set boundary conditions to be truncated and thus set improper boundary conditions. Conversion required changing the input format so that the boundary condition constant was read as two five-digit words instead of the previous ten-digit word. The scheme for extracting the boundary condition array values from the input constant was changed to operate on the two smaller words.

19. Subroutine Names

a) CDC allows alphanumeric identifiers up to eight characters in length.

b) IBM allows alphanumeric identifiers up to six characters in length. This required locating all of the subroutines with names greater than six characters and rewriting the identifiers with six character strings. The following list contains the subroutine names which were too long and the new six character names which were assigned to them.

BCRASHK	changed to	BCRSBK
BCZASHK		BCZSBK
BEXINIT		BEXINT
BFROMSI		BFRMSI
BOUNDRY		BOUNDY
OUTPUT1		OUTPT1
OVERISU		OVRISU
RUNDATA		RUNDAT
STARTUP		STRATUP

APPENDIX B

JOB CONTROL LANGUAGE (JCL) CARDS

A. USE IN THE OPERATING SYSTEM (OS)

Job control language (JCL) is the programming language which controls the IBM system 360 machine. It is the primary language, and other programming languages such as FORTRAN, ALGOL, COBOL, and so forth are used as sub-programs. The JCL program is concerned with setting up the machine to handle a specific program which is written in another language. This includes keeping track of accounting data, calling special programs such as the FORTRAN compiler, defining and constructing data sets, and terminating the job. A more detailed discussion of JCL is presented in reference 3.

B. REQUIRED CARDS

There are four basic types of JCL cards. The following are brief descriptions of their functions. A detailed explanation of their functions is contained in reference 3.

1. Job Card

The job card is the first card of the program and contains all the accounting data for the particular job.

2. EXEC Card

This is the second card of the program and contains instructions for execution of the program. This card can be used to call a catalogued procedure or a program stored

in a private library for execution. It is also used to specify various parameter options which are available in catalogued procedures. REGION, or core storage, requirements can also be specified on this card.

3. Data Definition (DD) Card

This card follows the EXEC card and is used to set up any data sets that may be required by the program. It specifies such parameters as the data set identifiers, device on which the data set will reside, amount of memory which will be required for the data set, disposition of the data set, and the data control block (DCB) which describes the construction of the data set.

4. Delimiter (/*) Card

This card limits the extent of a particular operation. It is placed at the end of a set of cards which are being read for a particular operation in the program and signals the end of that particular set. For instance, when a source program is compiled, the /* card is placed at the end of the source deck and signals the end of the cards which are to be compiled. If data cards are to be input for execution of the source program, they are placed between a DD card and another /* card which this time signals the end of the input data set.

C. CATALOGUED PROCEDURES

1. Descriptions and Use

Catalogued procedures are programs which are written and placed in the machine library for use by the general user. These programs are catalogued because they are very common programs and are used frequently by many users. Each catalogued procedure is named with an identifier. To call a specific procedure, its identifier is placed on the EXEC card following the EXEC. For example, the EXEC card

```
// EXEC FORTC
```

would call the catalogued procedure FORTC which would then 'compile only' a FORTRAN program source deck which followed in the input stream.

2. Over-ride Procedures

Since the catalogued procedure is a program which is already written, all available options are preset. These may be modified by over-riding the particular preset option as described below.

a. Time Over-Ride

The general user at NPS is currently allowed 20 seconds for his job. This includes compilation and execution. It does not include the time used in reading or writing on data sets. This 20 second time may be over-ridden by specifying the required time on the JOB card following the 'name' field. For example, the following

JOB card

//YAG30833 JOB (0833,0739FP,FA11),'THESIS',TIME=10
would allocate ten minutes to job YAG30833.

b. REGION Over-Ride

The general user is allowed 100K bytes of magnetic core for his job. This includes space for the program and any external subroutines used by the main program. The REGION over-ride is made on the EXEC card. For example the EXEC card

// EXEC FORTCLG,REGION.FORT=150K
would allocate 150K bytes of core to the FORT step in the catalogued procedure FORTCLG.

c. Data Set Space Over-Ride

The space allocation varies with the data set. For example the general user is allowed three cylinders of 2314 disc storage for data set FT06F001, which is the data set for the line printer. If more space is required, the over-ride is placed in the SPACE parameter of the DD statement for that data set. For example the statement

//GO.FT06F001 DD SPACE=(CYL,(3,1))
would initially allocate three cylinders of disc storage to the data set. If this were not sufficient, additional space would be allocated one cylinder at a time until a total of 18 cylinders is used.

d. Parameter (PARM) Over-Ride

Standard options are set in a catalogued procedure when it is written. These are outlined in a

description of the procedure. If any of these options need to be modified, they may be over-ridden on the EXEC card. For example the catalogued procedure FORTCLGP has the standard options MAP and NODECK in addition to a list of others. The MAP option prints out a map showing the location of all identifiers used in the program. NODECK indicates that no object deck is to be punched. If the operator wants the procedure to punch an object deck in addition to the normal procedure, the following card

```
// EXEC FORTCLGP,PARM.FORT='MAP,DECK'
```

would accomplish this operation. The standard compiler option is NOMAP. Procedure FORTCLGP has changed this to MAP. Note that it was necessary to specify MAP again in the over-ride statement. When one member of the parameter list is over-ridden, all of the other members revert to the standard options, so MAP would have become NOMAP if it had not been respecified.

3. QUICKRUN

QUICKRUN is a special implementation of the catalogued procedure FORTCLG. It is available to general users whose jobs run 20 seconds or less, use 100K bytes of core or less, do not use special peripheral equipment, and do not call specialized library subroutines.

The procedure reads, compiles, and executes a FORTRAN source program deck. The output suppresses all page skips unless the page skip suppress is over-ridden. The advantage

of a QUICKRUN job is its fast turn-around time of about five minutes since these jobs have priority over all other classes of jobs. This makes QUICKRUN an ideal procedure to use for testing or debugging small programs or sub-routines. The JCL deck setup for a QUICKRUN job is listed in paragraph D of this appendix.

D. USEFUL PROGRAMS

The following JCL programs were used in the conversion of Lindemuth's code. Each program is described briefly and is then listed as it would appear for execution. The examples all use the author's accounting data. The reader must insert his own accounting data where appropriate. Reference 3 fully describes the required accounting data.

In these programs the option 'MAP' refers to the listing of the contents of storage areas by identifier names. This option is standard on all of the catalogued procedures.

The option 'LIST' causes the object deck listing to be printed out. This listing is useful in debugging because it associates relative storage addresses with statement numbers in the source program.

THIS PROGRAM WILL COMPILE THE SOURCE PROGRAM DECK FOLLOWING
THE //FORT.SYSIN DD * CARD AND WILL PRODUCE AN
OBJECT DECK.

```
//YAG30833 JOB (0833,0739FP,FA11),'CHARLIE YAGER THESIS'  
// EXEC FORTCD  
//FORT.SYSIN DD *  
SOURCE PROGRAM DECK.  
/*
```

THIS PROGRAM WILL COMPILE THE SOURCE PROGRAM DECK FOLLOWING
THE //FORT.SYSIN DD * CARD AND WILL ADD A PRINTED LISTING
OF THE OBJECT DECK MAP.

```
//YAG30833 JOB (0833,0739FP,FA11),'CHARLIE YAGER THESIS'  
// EXEC FORTC,PARM.FORT='MAP,LIST'  
//FORT.SYSIN DD *  
SOURCE PROGRAM DECK.  
/*
```

THIS PROGRAM WILL COMPILE THE SOURCE PROGRAM DECK FOLLOWING
THE //FORT.SYSIN DD * CARD AND WILL PROVIDE BOTH THE
OBJECT DECK AND ITS MAP LISTING.

```
//YAG30833 JOB (0833,0739FP,FA11),'CHARLIE YAGER THESIS'  
// EXEC FORTCD,PARM.FORT='MAP,LIST,DECK'  
//FORT.SYSIN DD *  
SOURCE PROGRAM DECK.  
/*
```

THIS PROGRAM WILL COMPILE, LINK EDIT, AND EXECUTE THE SOURCE
PROGRAM FOLLOWING THE //FORT.SYSIN DD * CARD.

```
//YAG30833 JOB (0833,0739FP,FA11),'CHARLIE YAGER THESIS'  
// EXEC FORTCLG  
//FORT.SYSIN DD *  
SOURCE PROGRAM DECK.  
/*  
//GO.SYSIN DD *  
DATA DECK IF REQUIRED.  
/*
```

THIS PROGRAM WILL COMPILE, LINK EDIT, AND EXECUTE THE SOURCE
PROGRAM FOLLOWING THE //FORT.SYSIN DD * CARD AND WILL
PUNCH AN OBJECT DECK.

```
//YAG30833 JOB (0833,0739FP,FA11),'CHARLIE YAGER THESIS'  
// EXEC FORTCLGP,PARM.FORT='MAP,DECK'  
//FORT.SYSIN DD *  
SOURCE PROGRAM DECK.  
/*  
//GO.SYSIN DD *  
DATA DECK IF REQUIRED.  
/*
```


THIS PROGRAM WILL COMPILE, LINK EDIT, AND EXECUTE THE SOURCE
PROGRAM FOLLOWING THE //FORT.SYSIN DD * CARD AND WILL
PROVIDE AN OBJECT DECK MAP LISTING.

```
//YAG30833 JOB (0833,0739FP,FA11),'CHARLIE YAGER THESIS'  
// EXEC FORTCLG,PARM.FORT='MAP,LIST'  
//FORT.SYSIN DD *  
SOURCE PROGRAM DECK.  
/*  
//GO.SYSIN DD *  
DATA DECK IF REQUIRED.  
/*
```

THIS PROGRAM WILL COMPILE, LINK EDIT, AND EXECUTE THE SOURCE
PROGRAM FOLLOWING THE //FORT.SYSIN DD * CARD AND WILL
PROVIDE BOTH THE OBJECT DECK AND ITS MAP LISTING.

```
//YAG30833 JOB (0833,0739FP,FA11),'CHARLIE YAGER THESIS'  
// EXEC FORTCLGP,PARM.FORT='MAP,LIST,DECK'  
//FORT.SYSIN DD *  
SOURCE PROGRAM DECK.  
/*  
//GO.SYSIN DD *  
DATA DECK IF REQUIRED.  
/*
```

THIS PROGRAM WILL LINK EDIT, AND EXECUTE THE SET OF OBJECT
DECKS FOLLOWING THE //LINK.SYSIN DD * CARD.

```
//YAG30833 JOB (0833,0739FP,FA11),'CHARLIE YAGER THESIS'  
// EXEC FORTLG  
//LINK.SYSIN DD *  
SET OF OBJECT DECKS.  
/*  
//GO.SYSIN DD *  
DATA DECK IF REQUIRED.  
/*
```

THE FOLLOWING PROGRAM WILL COMPILE, LINK EDIT AND EXECUTE
THE SOURCE PROGRAM FOLLOWING THE JOB CARD USING THE QUICKRUN
FACILITY. NOTE THAT THE JOB CARD IS OF A DIFFERENT FORMAT

```
// QUICKRUN  
#YAG30833 FORTGO (0833,0739FP,FA11),'QUICKRUN EXAMPLE'  
SOURCE PROGRAM DECK.  
# DATA  
DATA DECK IF REQUIRED.  
//
```


THIS PROGRAM WILL LIST THE CARDS FOLLOWING
THE //SYSUT1 DD DATA,DCB=BLKSIZE=80 CARD.

```
//YAG30833 JOB (0833,0739FP,FA11),'CHARLIE YAGER THESIS'  
// EXEC PGM=IEBGENER  
//SYSPRINT DD SYSOUT=A  
//SYSIN DD DUMMY  
//SYSUT2 DD SYSOUT=A,DCB=(RECFM=F,BLKSIZE=80)  
//SYSUT1 DD DATA,DCB=BLKSIZE=80  
    INPUT DATA DECK.  
/*
```

THIS PROGRAM WILL COMPILE THE SOURCE PROGRAM DECK FOLLOWING
THE //FORT.SYSIN DD * CARD.

```
//YAG30833 JOB (0833,0739FP,FA11),'CHARLIE YAGER THESIS'  
// EXEC FORTC  
//FORT.SYSIN DD *  
    SOURCE PROGRAM DECK.  
/*
```


APPENDIX C

FILES

A. DEFINITIONS

A distinction should be made between FILES and DATA SETS. This paper will use the following descriptions:

1. File

A file is a collection of data or information which is used for a specific purpose. This collection will be given a characteristic identifier name which can be used for reference. For example, the file to hold all the data generated by the program is given the identifier DATA.

2. Data Set

A data set is the device from which or to which data and information is passed. Data sets are defined in JCL cards which are not part of the source program code. Common data sets are the card reader, card punch, line printer, magnetic tapes, magnetic discs, and paper tapes. See appendix H for a detailed description of data sets and their use.

B. DESCRIPTIONS

1. DATA

This file consists of all the input data cards used to start or restart a program. In its present form the program will use this file with data set FT05F001, the card reader. Appendix E describes all of the input data cards used by this program.

2. MHDOUT

This file consists of all the output information generated during execution of the program. In its present form this program will use this file with data set FT06F001, the line printer.

3. RECORD

This file consists of all the information necessary to restart a program plus the values of each variable at each mesh point at each time step. Figure 1 is a graphical description of the organization of the file. In its present form this program will use this file with data set FT10F001, a data set on disc storage.

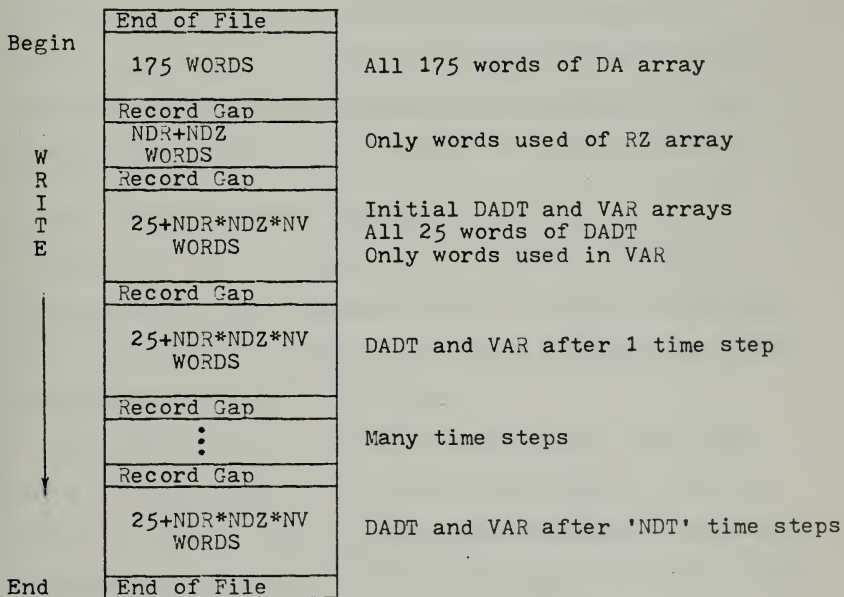


Figure 1. Organization of File DATA

4. OLDREC

This file is identical to RECORD. OLDREC is used in the restart mode of operation and is created from RECORD by changing the name of RECORD to OLDREC. When the program is operated in the restart mode, OLDREC is used by subroutine STRTUP as a source of input (note that this is not a part of file DATA) and the program will enter the new computed values into a new file RECORD. In its present form this program will use this file with data set FT30F001, a data set on disc storage.

5. SCRAT

This file is a work area for subroutines CHECK and SPECHK and is created only when these diagnostic routines are used. In its present form this program will use this file with data set FT40F001, a data set on disc storage which is destroyed after each run of the program.

C. USE WITH READ/WRITE COMMANDS

Data sets are written onto and read from their respective units by the standard FORTRAN statements READ and WRITE. Information buffered in and out may take two forms, either format controlled or binary.

The format controlled READ statement takes the form READ(a,b) list, where 'a' is the device number (which is identical to the integer in the third and forth positions of the data set number), 'b' is the format statement number, and 'list' is the set of variables to be read.

The binary READ statement which has no format control, takes the form READ(a) list where 'a' is the device number and 'list' is the set of variables to be read. The binary READ/WRITE is used to transfer information between the data set and core without changing the bit pattern.

Files DATA and MHDOUT use format controlled I/O statements while files RECORD, OLDREC, and SCRAT use binary I/O statements.

This program uses integer identifiers for the device number in all of the I/O statements. The identifiers are set to the appropriate integer values in the initial steps of the main program. This allows flexibility in the choice of data sets in that a data set can be altered by changing one card in the main program instead of changing every READ/WRITE card in the entire program. Table II is a reference of files and data sets including all identifiers which are used by the program in its present form.

TABLE II

DATA SET IDENTIFICATION REFERENCE

FILE NAME	DATA SET ID	DEVICE	VARIABLE ID	USED IN SUBROUTINE(S)
DATA	FT05F001	CARD READER	INP	MAIN,BEXINT,INIT4,INIT5, MATRIX,MESH,STRUP,TCINIT
MHDOUT	FT06F001	LINE PRINTER	IOUT2	MAIN,CHECK OUTPT1,PLOTR, RBC,RUNDAT,SPECHK,TCINIT, TEPLOT,ZBC
RECORD	FT10F001	DISC	IOUT1	MAIN,STRUP
OLDREC	FT30F001	DISC	INP2	STRUP
SCRAT	FT40F001	DISC	IOUT4, T1	CHECK

APPENDIX D

SUBROUTINE DESCRIPTIONS

This section will list all of the subroutines included in the present code and will briefly describe their functions and operations. The subroutine functions can be divided into five categories.

1. LOGIC AND CONTROL: These are the subroutines which primarily control the operation of the code by directing the call of other subroutines after manipulation of control variables. Subroutines included in this category are MAIN, BCRETS, BCZOTS, BCRSHK, BCZSHK, BOUNDY, and OVRISU.

2. INITIALIZATION: These are the subroutines which set up all of the initial values required to run a particular problem. Except for three logic control cards, all of the input data cards are read by the subroutines in this category. Subroutines included in this category are BEXT, INIT1, INIT4, INIT5, MATRIX, MESH, SETUP, STRTUP, and TCINIT.

3. OPERATION: These are the subroutines which perform the actual calculations during execution. Subroutines included in this category are BCR, BCZ, BFRMSI, MAT2, ONESID, SHOCK, TRANCO, and TRIANG.

4. OUTPUT: These are the subroutines which process information for inclusion in the output files RECORD and MHDOUT. While much of their processing is done before

any calculation takes place, their operations are still considered output. Subroutines included in this category are BEXINT, BUFFER, OUTPT1, PLOT, RUNDAT, TEPLT, and ZBC.

5. DIAGNOSTIC: These subroutines are used to diagnose numerical operations. By performing inverse operations on calculated matrices, variations in the bit patterns of words can be observed and analyzed. The subroutines in this category are CHECK and SPECHK.

The following list provides a brief description of the function of each subroutine.

A. MAIN PROGRAM

This is one of the logic and control routines and controls the general operation of the entire program. It reads cards number 1, 2, and 3 from the input data set to control diagnostics requests and data retention in file RECORD. This routine also checks for negative density, ion temperature and electron temperature. If these quantities occur, the program will terminate and print out that negative values were found during execution in a particular time step.

B. BCR (Boundary Conditions for R)

This is one of the operation subroutines and is called by MAIN and BOUNDY. BCR sets up boundary matrices and vectors along the R_1 and R_{NDR} lines corresponding to

equations 3.5-3 and 3.5-4 of reference 1. Boundary condition control parameters are passed from the initializing subroutines through COMMON blocks C4 and C6 and determine which of a set of pre-defined boundary conditions will be applied to a particular variable. Subroutine RBC forms printed statements of the selected boundary conditions.

C. BCRETS (Boundary Conditions for R, Even Time Step)

This is one of the control and logic subroutines and is called by MAIN. It calls RBC through BOUNDY and is concerned with boundary calculations corresponding to equations 3.5-3 and 3.5-4 of reference 1 with 'n' replaced by 'n+1' and 'n+1' replaced by 'n+2'. Again operations are on lines R_1 and R_{NDR} .

D. BCRSBK (Boundary Conditions for R, after Shock Smoother)

This is one of the control and logic subroutines and is called by MAIN. It calls RBC through BOUNDY and is concerned with boundary calculations along line R_1 and R_{NDR} . BCRSHK causes boundary values to be readjusted after shock smoothing if used.

E. BCZ (Boundary Conditions for Z)

This is one of the operation subroutines and is called by MAIN and BOUNDY. BCZ sets up boundary matrices and vectors along the Z_1 and Z_{NDZ} lines corresponding to equations 3.5-5 and 3.5-6 of reference 1. Boundary condition control parameters are passed from the initializing subroutines through COMMON blocks C4 and C6 and determine

which of a set of pre-defined boundary conditions will be applied to a particular variable. Subroutine ZBC forms printed statements of the selected boundary conditions.

F. BCZOTS (Boundary Conditions for Z, Odd Time Step)

This is one of the control and logic subroutines and is called by MAIN. It calls ZBC through BOUNDY and is concerned with boundary calculations corresponding to equations 3.5-5 and 3.5-6 of reference 1 with 'n' replaced by 'n+1' and 'n+1' replaced by 'n+2'. Again operations are on lines Z_1 and Z_{NDZ} .

G. BCZSHK (Boundary Conditions for Z, after SHock Smoothing)

This is one of the control and logic subroutines and is called by MAIN. It calls ZBC through BOUNDY and is concerned with boundary calculations along lines Z_1 and Z_{NDZ} . BCZSHK causes boundary values to be readjusted after shock smoothing if used.

H. BEXINT(J,K) (B Field External, INiTially)

This is one of the output subroutines and forms the labels for the external B field values which are read in on input card number 10. Data values are passed to other subroutines through COMMON/C14/.

I. BEXT (B Field External)

This is one of the initializing subroutines and works in conjunction with BEXINT to provide a uniform external B field. BEXT is used when $IBE \neq 0$. See section 3.8, page 83 of reference 1. Initial values are passed to BEXT

through COMMON/C14/ and the calculated values are returned to other subroutines through COMMON blocks C11 and C26.

J. BFRMSI(J,K,BR,BZ) (B Field FRom PSI)

This is one of the operation subroutines and calculates the B field components BR and BZ from the stream function ψ . BFRMSI is called from SETUP and is used only when anisotropic thermal conductivities are used (see page 29 of reference 1).

K. BOUNDY(IENT, NNNN NNN, ID) (BOUNDary)

This is one of the control and logic subroutines and works in conjunction with BCRETS, BCRSHK, BCZOTS, and BCZSHK to perform boundary calculations corresponding to equations 3.5-5 and 3.5-6 of reference 1.

L. BUFFER(IROR,IARRAY,ITP)

This is one of the output subroutines and controls the transfer of data into and out of file RECORD. The data set for file RECORD is defined for logical records of 100 bytes which corresponds to a list of 25 words. The arrays DA, RZ, and VAR are all longer than 25 words and require additional control to be read into file RECORD. This is the function of BUFFER.

The argument IROR for READ or RITE) specifies whether the subroutine is to READ from or WRITE into the data set. If IROR=0, BUFFER READS. If IROR=1, BUFFER WRITES.

IARRAY specifies which of three arrays are to be used with the I/O command. IARRAY=1 specifies array DA, IARRAY=2 specifies array RZ, and IARRAY=3 specifies array VAR.

ITP is the device identifier for the data set and is included in the argument list as the variable ITP. Its value is specified elsewhere in the program.

M. CHECK

This is one of the diagnostic subroutines and works in conjunction with SPECHK to print internal diagnostics. These diagnostics are used to determine which terms in the difference equations are responsible for non-physical values. Use of CHECK and SPECHK is determined by input parameters on cards 1, 2, and 13 of the input data set. Examples of the output from CHECK and SPECHK are included in reference 9.

N. INIT1 (INITializing Routine Number 1)

This is one of the initializing subroutines and sets up the VAR array at time $t=0$. This subroutine sets up a uniform plasma such as that used in theta-pinch studies. This problem is discussed in chapter 5 of reference 1.

O. INIT4 (INITializing Routine Number 4)

This is one of the initializing subroutines and sets up the VAR array at time $t=0$. This subroutine sets up a high density center plasma surrounded by a low density background such as that used in laser produced plasma

studies. This problem is discussed in chapter 6 of reference 1. Initial condition parameters are read on card number 12A of the input data set.

P. INIT5 (INITializing Routine Number 5)

This is one of the initializing subroutines and sets up the VAR array at time $t=0$. This subroutine sets up a discontinuous plasma such as that used in hydrocode test problems. This problem is discussed in chapter 4 of reference 1.

Q. MAT2(J,K)

This is the primary operation subroutine of the program. It determines matrices A, B, and C and the vector V which appear in difference equations 3.2-1, 3.5-1, and 3.5-2 of reference 1. The matrices A', B', and C' and vector V' (3.1-13, 3.3-12, and 3.3-25 of reference 1) are calculated and then converted to A, B, C, and V. The code is set up as if all ten components of W are to be calculated. However, if IV(K) is greater than NV, then the Kth component of W is not being calculated. MAT2 is called by MAIN and passes information back and forth through numerous COMMON blocks as described in appendix F.

R. MATRIX

This is one of the initializing subroutines and is the first one called by subroutine STRTUP in the start-up mode. It reads cards number 6, 7, and 8 from the input data set. The important control parameters in this subroutine are JAV and JVERS.

1. JAV: JAV determines which alternate variables are to be used, if any. An alternate variable is one which can be determined from a component of the vector W by either multiplying or dividing by the density. JAV is the parameter in a computed GO TO statement which branches to different values for the integer constant IIAV. IIAV is later used to set the values in the vector IAV which is then passed to other subroutines through COMMON/C6/. Figure 1 lists the values available for JAV and the function performed for each.

2. JVERS: JVERS determines a preselected set of program options. Included in each set is the number of dimensions, the number of variables, the variables to be calculated and their order in the vector U, and the coordinate system to be used in the calculation. Figure 1 lists the values available for JVERS and the function performed for each.

3. Other Parameters: The integer constants JJDDR and JJDDZ are read and used to assign values to the JDDR and JDDZ arrays which are then passed to other subroutines through COMMON/C22/. Appendix E lists the input parameters read by MATRIX.

MATRIX also forms messages in the IDA and ADDA vectors which are printed by RUNDAT. These messages list the input parameters read by MATRIX.

JAV	IIAV	Alternate Variables
1	0	No Alternate Variables
2	1000110000	SV=1/RO . . . PI=TI*RO, PE=TE*RO
3	1 SB=SI/R2=AP/R
4	110000 PI=TI*RO, PE=TE*RO
5	2 AP
6	110002 PI=TI*RO, PE=TE*RO . . . AP
7.	110001 PI=TI*RO, PE=TE*RO . . . SB=SI/RO

JVERS	DIM	ICoord	U Vector
0	2	0	(RO,VR,TI,TE,BZ)
1	2	0	(RO,VR,TI,TE,BP,BZ)
2	1	0	(RO,VR,TI,TE,BZ)
3	1	0	(RO,VR,TI,TE,BP,BZ)
4	1	0	(RO,VR,TI,TE)
5	2	0	(RO,VR,VP,VZ,TI,TE,BP,SI)
6	2	0	(RO,VR,VZ,TI,TE,SI)
7	1	0	(RO,VR,TI,TE,BP,SI)
8	1	0	(RO,VR,TI,TE,SI)
9	2	0	(RO,VR,VZ,TI,TE)
10	1	1	(RO,VR,TI,TE)
11	1	-1	(RO,VR,TI,TE)
12	1	-1	(RO,VR,TE)
13	1	1	(RO,VR,TE)
14	1	0	(RO,VR,TE)

Figure 1. MATRIX Control Parameter List

Figure 1 is to be used as a reference for the options which are presently available in subroutine MATRIX. The first block indicates what value of IIAV is assigned by the code for each value of JAV read in on input card number 7. For instance if JAV=2 is read in on card number 7, the code will set IIAV=100011000. This indicates that the first, fifth, and sixth components of the vector W will be treated as alternate variables. The effect of this is shown in the last column. The alternate variables are indicated and the '.' indicates that the variable is unchanged. Thus if JAV=2, the vector W would become $W=(SV,VR,VP,VZ,PI,PE,BR,BP,BZ,SI)$ instead of $W=(RO,VR,VP,VZ,TI,TE,BR,BP,BZ,SI)$.

The second block indicates which dependent variables will be calculated for each value of JVERS read in on input card number 7. For instance if JVERS=6, the code would calculate (in two dimensions and cylindrical coordinates) the dependent variables RO,VR,VZ,TI,TE, and SI. The other components of W would not be calculated. This assumes that JAV=0. However, if JAV=2 while JVERS=6, the dependent variables which would be calculated are SV, VR, VZ, PI, PE, and SI.

S. MESH

This is one of the initializing subroutines and sets up the mesh array RZ. Provisions are made for three different meshes in each dimension. Each mesh can be either uniformly spaced, or spaced geometrically. The code is limited to rectangular meshes set up in a rectangular domain. The parameters on card(s) 11A (and) 11B determine the mesh to be used.

T. ONESID (ONE-SIDed Derivatives)

This is one of the operation subroutines and changes standard centered first derivatives (equation 3.1-20 of reference 1) to one-sided derivatives (equations 3.8-7 and 3.8-19). The dependent variables which are to use the one-sided derivatives are determined by the JDDR and JDDZ arrays which themselves were determined by the input parameters JJDDR and JJDDZ respectively.

U. OUTPT1 (OUTPuT Routine Number 1)

This is the primary output subroutine in the program. It prints out the values of each dependent variable being calculated at each mesh point for each time step as determined by NDTPNT. Plots of the dependent variables are presented as well as the axial and radial distances at each mesh point. OUTPT1 is called by MAIN through OVRISU to print out when required.

V. OVRISU

This is one of the logic and control subroutines and serves merely to call other subroutines. It replaces

some of the overlays which were used in the original CDC 3400 version of the code.

W. PLOTR

This is one of the output subroutines and is used in conjunction with TEPLLOT to plot the Z line values of TE (electron temperature) vs. R. Its coding is similar to the plotting coding used in OUTPT1.

X. RBC (R Boundary Conditions)

This is one of the output subroutines and forms a part of the boundary condition messages which appear on the first page of the output printout. RBC is called by MAIN through OVRISU.

Boundary condition information is passed to RBC through vectors IBCRR and IBCR in COMMON/C4/. Vector IIV in COMMON/C1/ provides information concerning the variables to be calculated and the order in which they appear in the vector U. These control parameters determine the message which is buffered out to the output data set.

This subroutine has no effect on the dependent variables and could be eliminated if hard copy of the boundary conditions is not required.

Y. RUNDAT (RUN DATA)

This is one of the output subroutines and produces the listing of initial conditions and boundary conditions to be listed on the first page of the output printout. The data which is printed out is stored in arrays IDA and

ADDA by many other subroutines during initialization procedures.

Z. SETUP(J,K)

This is one of the initialization subroutines and sets up the arrays in COMMON blocks C8, C9, C10, C11, C15, C25, C26, and C27 which are used by MAT2. Transport coefficients are calculated for the mesh point NJ,KE and its eight neighbors by calling TRANCO. Storage limitations required computing the transport coefficients three times during each time step. On machines with sufficient core, the transport coefficients at all mesh points could be calculated prior to beginning a time step and thus save much computation time in this subroutine.

AA. SHKMAT

At the present time this is a dummy subroutine.

BB. SHOCK(JJJ,KKK)

This is one of the operation subroutines and does shock smoothing as discussed in section 3.6 of reference 1. It is called by MAIN and passes its data in and out through COMMON/C7/.

CC. SPECHK (SPEcial CHeck)

This is one of the diagnostic subroutines and works in conjunction with CHECK to print internal diagnostics. These diagnostics are used to determine which terms in the difference equations are responsible for non-physical values. Use of CHECK and SPECHK is determined by input

parameters on cards 1, 2, and 13 of the input data set. Examples of the output from CHECK and SPECHK are included in reference 1.

DD. STRTUP(ISUB,IPU,JJ,KK) (StArTUP)

This is one of the initializing subroutines and is the first one called by subroutine OVRISU in the startup mode. It reads cards number 4 and 5A or 5B from the input data set. All initialization is controlled by STRTUP as determined by the input cards. The argument list is used to pass values back to the calling program.

STRTUP can be used to start a new run or to restart a previous run which had been terminated.

1. Startup Mode: In this mode STRTUP reads cards number 4 and 5A. It then fills the boundary condition and ISHK vectors as specified by input parameters on card 5A, calls subroutines MATRIX, TCINIT, and BEXINT (if specified), and reterns the values of NDTPT, IPU, 1, and NDTPT-1 to the calling program through the argument list.

2. Restart Mode: The code provides two restart capabilities. The first will pick up execution at the point where the previous execution was terminated manually before NDT steps were completed. This type of termination requires the use of sense switch number one. Since the sense switch facility is not available, this restart mode will not be used at this time.

3. The second restart mode picks up execution at any previous time step in any previous file. The file from which the execution is to continue must be placed in data set FT30F001 before the program is executed. Card number 5A need only contain values for IPU, IFROM, and IST. Setting IFROM not equal to zero will cause the code to read from data set FT30F001 the vectors DA, RZ, and the DADT and VAR set which correspond to the time step specified by IST. Card number 5B will now be read, and the values can be changed at this point since this mode always starts a new file. If IPU equals zero, MATRIX is called and new parameters may be specified via card number 6. Otherwise an immediate return to OVRISU will be made and the program will continue with the same parameters that were used in the previous run.

EE. TCINIT(ITCDAT) (Transport Coefficient INITIALIZation)

This is one of the initializing subroutines and is the second one called by subroutine STRTUP in the startup mode. It reads card number 9 from the input data set. The primary control parameter is ITCDAT which is also the single argument in the subroutine parameter list.

When ITCDAT is equal to zero, (the start-up mode), data card number 9 is read. If ITCDAT is not equal to zero, (a restart mode) the data card is not read. Instead the relevant information is extracted from the ADDA vector and assigned to the proper identifiers. Execution then continues as in the start-up mode.

TCINIT evaluates some fundamental constants and the constants which are used in the transport coefficients (equation 2.3 of reference 1).

This subroutine also forms messages in the IDA and ADDA vectors which are printed by RUNDAT. These messages list the input parameters read by TCINIT and some of the values computed during execution.

FF. TEPLOT

This is one of the output subroutines and is used in conjunction with PLOTR to plot the Z line values of TE (electron temperature) vs. R. It is used to manually monitor TE for oscillations. If required, this can be easily modified to include all the dependent variables.

GG. TRANCO(VARI,TC,DTCDV,IENT,BE,TCD) (TRANsport COefficients)

This is one of the operation subroutines and computes the values of the transport coefficients which are discussed in section 2.3 of reference 1. Constants in the transport coefficients are passed to TRANCO from TCINIT through COMMON/C3/, and the alternate variable designators are passed through COMMON/C6/.

Dependent variable values are passed from the calling subroutine through the VARI vector in the subroutine argument list. Computed transport coefficients are passed back to the calling routine through the vectors TC, TCD, and the array DTCDV in the subroutine argument list. External B field values are also passed to TRANCO through

the subroutine argument list in the BE vector. The other identifier in the argument list, IENT, is a control identifier which, if set to one, skips the coding for transport coefficient derivatives and leaves the DTCDV array equal to zero.

HH. TRIANG(IF1)

This is one of the operation subroutines and solves the equations $A U = V$, and $A M = B$ by Gaussian elimination. The subroutine argument IFu determines which equation is to be solved. If IF1=1, then only the first equation is solved. If IF1 is not equal to one, then both equations are solved.

The arrays A and B and the vector V are passed to TRIANG through COMMON/C7/ and the control parameter NV is passed through the DA vector of COMMON/C3/.

The array M and the vector U are passed back to the calling subroutine through COMMON/C7/. Values of U are returned in V and values of M are returned in B.

These arrays are discussed in section 3 of reference 1.

II. ZBC (Z Boundary Conditions)

This is one of the output subroutines and forms a part of the boundary condition messages which appear on the first page of the output printout. ZBC is called by MAIN through OVRISU.

Boundary condition information is passed to ZBC through vectors IBCZZ and IBCZ in COMMON/C4/. Vector IIV

in COMMON/C1/ provides information concerning the variable to be calculated and the order in which it appears in the vector U. These control parameters determine the message which is buffered out to the output data set.

This subroutine has no effect on the dependent variables and could be eliminated if hard copy of the boundary conditions is not required.

	MAIN	BCR	BCRETS	BCRSHK	BCZ	BCZOTS	BCZSHK	BEXINT	BEXT	BFRMSI	BOUNDY	BUFFER	CHECK	INIT1	INIT4	INIT5	MAT2	MATRIX	MESH	ONESID	OUTPT1	OVRSU	RBC	RUNDAT	PLOTR	SETUP	SHKMAT	SHOCK	SPECHK	STRTUP	TCINIT	TEPLOT	TRANCO	TRIANG	ZBC
MAIN		•	•	•	•	•						•	•				•					•				•	•	•					•		
BCR																																			
BCRETS												•																					•		
BCRSHK												•																							
BCZ																																			
BCZOTS												•																							
BCZSHK												•																							
BEXINT																																			
BEXT																																			
BFRMSI																																			
BOUNDY		•			•																													•	
BUFFER																																		•	
CHECK																													•						
INIT1																						•													
INIT4																						•													
INIT5																						•													
MAT2																																			
MATRIX																																			
MESH																																			
ONESID																																			
OUTPT1																																			
OVRSU														•	•	•			•		•		•	•						•	•			•	
RBC																																			
RUNDAT																																			
PLOTR																																			
SETUP												•	•																					•	
SHKMAT																																		•	
SHOCK																																			
SPECHK																																			
STRTUP																																		•	
TCINIT																																			
TEPLOT																																			
TRANCO																																			
TRIANG																																			
ZBC																																			

Figure 2. Cross-reference of Subroutines

To find all of the subroutines called by a particular subroutine, locate that subroutine in the left-hand column and read across.

To find all of the subroutines which call a particular subroutine, locate that subroutine in the top row and read down.

APPENDIX E

INPUT DATA CARD FORMATS

The following list includes all input cards which may be used in the program. They are not all required for any one run. The numbers assigned to these data cards are referenced in the subroutine descriptions of appendix D.

The description of each card includes the card number, statement number of the appropriate READ command, and the card format description. A typical statement number may be of the form 107-2. This indicates that the READ statement is not numbered and that it is located two cards in front of statement number 107 in the source code. Each line of the format description contains the card columns for a particular identifier, the identifier name, the identifier READ format, and a brief description of the identifier. More detailed descriptions of the identifiers are given in appendix I.

With a few exceptions discussed below, all of the input cards listed must be included in the input stream in the listed order, even though the parameters may all be zero. The following exceptions must be observed.

1. If the value of NREG on card number 11A is less than or equal to '1', card 11B must not be included in the input stream.

2. Cards 12A and 12B must not be read together.

These cards are input to specific initializing subroutines and only one of these subroutines is used during any one run. Therefore the input card for the subroutine which is not being used must be removed from the input stream.

3. Card number 13 may be many cards with the same format. This card specifies what action is to be taken when a diagnostic call is made. There must be one of these 'instruction' cards for each mesh point that is specified on card number 2. As a check, the number of input cards number 13 will be the same as N2BCHK which is read on card number 1. If N2BCHK=0, then no cards number 13 are required in the input stream.

CARD NR 1 READ AT STATEMENT NR 1-1 OF MAIN
 1-2 N2BCHK (12) NUMBER TO BE CHECKED (MUST BE LE 20)
 3-80 BLANK

CARD NR 2 READ AT STATEMENT NR 2-1 OF MAIN
 1-2 J2BCHK(1) (12) J TO BE CHECKED
 3-4 K2BCHK(1) (12) K TO BE CHECKED
 5-6 J2BCHK(2) (12)
 7-8 K2BCHK(2) (12)
 •
 •
 K2BCHK(N) (12) N=N2BCHK
 J2BCHK(N) (12) N=N2BCHK
 21-80 BLANK

CARD NR 3 READ AT STATEMENT NR 3-1 OF MAIN
 1-3 NDTBUF (13) NUMBER OF THE TIME STEP TO BE BUFFERED OUT.
 4-80 BLANK

CARD NR 4 READ AT STATEMENT NR 22-1 OF STRUP
 1-5 DATED JULIAN DATE .. IE 72138 FOR 138TH DAY OF 1972

6-10 BLANK
 11-15 ITAPE
 16-20 BLANK
 21-25 IFYL
 26-80 BLANK

CARD NR 5A READ AT STATEMENT NR 66-2 OF STRUP
 1 IPLAVC
 2 IPU
 3 IVERS
 4 IPIVOT
 5 ISTEPO
 6-10 NDI
 11-15 NDI
 16-20 NDI
 21-25 NDZ
 26-35 JBCRR
 36-45 JBCR
 46-55 JBCZ
 56-65 JBCZ
 66-75 ISHOCK
 76-77 IFROM
 78-80 IST

USED TO SELECT RESTART MODE

USED TO SET ISTEP ODD OR EVEN
 NUMBER OF TIME STEPS TO BE PRINTED
 NUMBER OF MESH POINTS IN THE R DIRECTION
 NUMBER OF MESH POINTS IN THE Z DIRECTION
 INNER R BOUNDARY CONDITIONS
 OUTER R BOUNDARY CONDITIONS
 INNER Z BOUNDARY CONDITIONS
 OUTER Z BOUNDARY CONDITIONS

USED TO SELECT STARTUP MODE
 USED TO SELECT RESTART MODE

CARD NR 5B READ AT STATEMENT NR 107-2 OF STRUP
 1-2 BLANK
 3 IVERS
 4 IPIVOT
 5 BLANK
 6-10 NDI
 11-15 NDI
 16-25 BLANK
 26-35 JBCRR
 36-45 JBCR
 46-55 JBCZ
 56-65 JBCZ
 66-75 ISHOCK
 76-80 BLANK

NUMBER OF TIME STEPS TO BE PRINTED
 NUMBER OF TIME STEPS TO BE PRINTED

INNER R BOUNDARY CONDITIONS
 OUTER R BOUNDARY CONDITIONS
 INNER Z BOUNDARY CONDITIONS
 OUTER Z BOUNDARY CONDITIONS
 USED TO SELECT RESTART MODE

CARD NR	6	READ AT STATEMENT NR	2-1 OF MATRIX
1	1	IOVEX	SELECTS DIFFERENCE METHOD
2	2	MMAT	SELECTS CODING SUBSET IN MAT2
3	3	MATALT	SELECTS ISOTROPIC OR ANISOTROPIC THERMAL
4	4	ISO	CONDUCTIVITIES
5-10		BLANK	
11-20		DT	SIZE OF TIME STEP
21-30		CQA	SHOCK SMOOTHING
31-35		NZP(1)	NUMBER OF 1ST Z LINE TO BE PLOTTED BY OUTP11
35-40		NZP(2)	NUMBER OF 2ND Z LINE TO BE PLOTTED BY OUTP11
41-45		NZP(3)	NUMBER OF 3RD Z LINE TO BE PLOTTED BY OUTP11
45-55		JDDR	ONE-SIDED R DERIVATIVES
56-65		JDDZ	ONE-SIDED Z DERIVATIVES
66-70		NTCORV	NO TRANSPORT COEFFICIENT DERIVATIVE
71-80		BLANK	

CARD NR	7	READ AT STATEMENT NR	3-1 OF MATRIX
1-2	1	JVERS	SELECTS NUMBER OF VARIABLES
3	3	JAV	SELECTS PRESSURES INSTEAD OF TEMPERATURES
4	4	INICON	INITIAL CONDITION SELECTOR
5	5	IBE	SEPARATES B FIELD INTO PLASMA AND EXTERNAL
6-10		BLANK	
11-20		ROINIT	PEAK DENSITY
21-30		TINIT	PEAK TEMPERATURE
31-40		ROMIN	PEAK MINIMUM DENSITY
41-50		TRO	IRRELEVANT
51-80		BLANK	

CARD NR	8	READ AT STATEMENT NR	40-1 OF MATRIX
1-5	1	BLANK	
6-15	6	BZINIT	INITIAL PLASMA B FIELD IN THE Z DIRECTION
16-25	16	BZMAX	MAXIMUM PLASMA B FIELD IN THE Z DIRECTION
26-35	26	BZF	FREQUENCY OF PLASMA B FIELD IN THE Z DIRECTION
36-45	36	BRAT	MIRROR RATIO
46-55	46	BPINIT	INITIAL PLASMA B FIELD IN THE PHI DIRECTION
56-65	56	BPMAX	MAXIMUM PLASMA B FIELD IN THE PHI DIRECTION
66-75	66	BPF	FREQUENCY PLASMA B FIELD IN THE PHI DIRECTION
76-80	76	BLANK	

CARD NR	9	READ AT STATEMENT NR	10-1 OF TCINIT
1-4		BLANK	
5		ITCVER	SUBROUTINE VERSION
6-15		CLNLM2	CONSTANT USED IN MODIFIED COULOMBIAN LOGARITHM
16-25		TCRO	TRANSPORT COEFFICIENT DENSITY
26-35		GAM	GAMMA FACTOR
36-45		PMASS	MASS OF PROTON OR ION
46-55		STREQO	SETS K EQUAL TO ZERO
56-65		(UNUSED)	
66-75		(UNUSED)	
76-80		BLANK	

CARD NR	10	READ AT STATEMENT NR	10-1 OF BEXINT
1-5		BLANK	
6-15		BZEINT	INITIAL EXTERNAL B FIELD IN THE Z DIRECTION
16-25		BZEMAX	MAXIMUM EXTERNAL B FIELD IN THE Z DIRECTION
26-35		BZEF	FREQUENCY EXTERNAL B FIELD IN THE Z DIRECTION
36-45		(UNUSED)	
46-55		(UNUSED)	
56-65		(UNUSED)	
66-75		(UNUSED)	
76-80		BLANK	

CARD NR	11A	READ AT STATEMENT NR	10-1 OF MESH
1-4		BLANK	
5		NREG	IF GT "1", CARD 11B IS READ
6-15		RMIN	MINIMUM RADIUS
16-25		RMAX	MAXIMUM RADIUS
26-35		ZMAX	MAXIMUM DISTANCE IN THE Z DIRECTION
36-45		PRAT	DETERMINES MESH SPACING FACTOR IN Z
46-55		ZRAT	DETERMINES MESH SPACING FACTOR IN Z
56-80		BLANK	

CARD NR	11B	READ AT STATEMENT NR	20-1 OF MESH
1-5		BLANK	
6-10		NDRI	(I5)
11-20		R1	(E10.3)
21-30		RRAT1	(E10.3)
31-35		NDZ1	(I5)
36-45		Z1	(E10.3)
46-55		ZRAT1	(E10.3)
56-80		BLANK	

CARD NR 12A READ AT STATEMENT NR 5-1 OF INIT4
 1-4 BLANK
 5 TOP
 6-15 SELECTS PROFILE OPTION
 16-25 PLASMA INITIAL BOUNDARY VELOCITY
 26-35 PLASMA INITIAL BOUNDARY RADIUS
 36-45 PLASMA BACKGROUND TEMPERATURE
 46-55 IRREVELANT
 56-65 MULTIPLE OF XO WHERE VELOCITY GOES TO ZERO
 66-75 IRREVELANT
 76-80 WHERE VELOCITY IS MAXIMUM
 BLANK

CARD NR 12B READ AT STATEMENT NR 5-1 OF INIT5
 1-4 BLANK
 5 ICYL
 6-15 XSO
 16-25 TRINIT
 26-35 ROINIT
 36-45 VLEFT
 46-55 NOT USED
 56-65 NOT USED
 66-75 NOT USED
 76-80 BLANK

CARD NR 13 READ AT STATEMENT NR 4-1 OF MAIN
 1 JCHK(1) =1 ARRAYS A,B,C,U ARE PRINTED
 2 JCHK(2) =1 ARRAYS E,F ARE PRINTED
 3 JCHK(3) =1 ARRAY U IS PRINTED
 4 JCHK(4) =1 CALCULATION OF E,F IS CHECKED
 5 JCHK(5) =1 CALCULATION OF VARIABLES AT NEW TIME STEP IS CHECKED

APPENDIX F

COMMON BLOCK STORAGE AND EQUIVALENCE

A. COMMON STORAGE

COMMON block storage is used to pass information between programs or subprograms without using a parameter list. This reduces the core storage required for the program because subprograms using the same named COMMON block use the same physical storage area in core.

In IBM FORTRAN, we are allowed one un-named COMMON block and as many named blocks as we need. Variables or arrays declared in an un-named COMMON block may not be assigned values in a DATA statement. However, variables or arrays in named COMMON blocks may be initialized in a DATA statement.

The named COMMON blocks are labeled by an alphanumeric identifier which may be up to six characters in length, the first of which must be an alphabetic character. The length of the COMMON block is determined by the first COMMON declaration. All COMMON declarations in subsequent subprograms must be less than or equal in length to the first declaration.

Because two or more subprograms use the same storage area, all the variables used and stored by one subprogram may be used by any other subprogram which has declared the same COMMON block. However, care must be taken to insure that the variables in the COMMON declaration list are in the same order as they are in other lists. For

example, if subprograms S1 and S2 both contain the statement 'COMMON/C1/ A,B,C' then the same identifier in each program would have the same value. However, if subprogram S2 contained the statement 'COMMON/C1/ A,C,B' then the value of A would be the same in both subprograms, B in S1 would be the same as C in S2, and C in S1 would be the same as B in S2.

B. EQUIVALENCE

EQUIVALENCE statements are used to assign variables or arrays in the same subprogram to the same physical area in storage. Again this reduces the core storage required for the program because many variables may occupy the same area in core storage. Variables in COMMON may not be equivalenced to each other. However, any number of variables not in COMMON storage may be equivalenced. As an example, consider a program which contains the following statements:

```
COMMON/C3/ A(5)
EQUIVALENCE (A(1),X),(A(2),Y(2)),(A(4),Z(1))
DIMENSION Y(3),Z(3)
```

As a result of this EQUIVALENCE statement, the following set will occupy the same storage location in core:

```
A(1) and Y(1) and X
A(2) and Y(2)
A(3) and Y(3)
A(4) and Z(1)
A(5) and Z(2)
```

Z(3) will be assigned a storage location of its own. Any time an operation modifies one member of an equivalenced

set, the other members will automatically be set to that value. For example, if the program set $X=1$, then $A(1)$ and $Y(1)$ would also contain the value 1.

C. USE

Lindemuth's program makes extensive use of COMMON blocks and EQUIVALENCE statements. Logical and control data is passed between subprograms via COMMON blocks. Some COMMON block lists are composed of identifiers, while others consist of large arrays which are then equivalenced to individual identifiers. The arrays are used to facilitate rapid buffering to the I/O devices. The data transferred through the arrays DA, RZ, DADT, and VAR are also placed on permanent storage devices for later use in restarting or data analysis.

D. REFERENCES

Figure 1 is a chart which cross-references subroutines with COMMON blocks used in the program. By reading across for a particular subroutine, the COMMON blocks used in that subroutine may be determined. Similarly by reading down for a particular COMMON block, the subroutines to which it is common may be determined.

Table III lists the COMMON blocks used in this program, the dimension of the block, and the variable list generally associated with the COMMON block. A notable exception to this list concerns subroutine MAT2. In COMMON blocks

C8, C9, C11, C15, and C23 the lists consist of single variables instead of the vectors listed in the other subroutines. These variables may be cross-referenced by referring to the appropriate COMMON declaration in the source listing for subroutine MAT2 which may be found in the program listing at the end of this paper.

Table IV lists the members of the DA and DADT vectors and the variables to which they are equivalenced.

	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C13	C14	C15	C20	C21	C22	C23	C25	C26	C27	C30
MAIN
BCR				
BCRETS						
BCRSHK					
BCZ					
BCZOTS					
BCZSHK					
BEXINT	
BEXT
BFRMSI	
BOUNDY					
BUFFER					
CHECK
INIT1					
INIT4	
INIT5	
MAT2
MATRIX	
MESH	
ONESID
OUTPT1
OVRI SU	
RBC	
RUNDAT	
PLOTR	
SETUP	
SHKMAT	
SHOCK	
SPECHK
STRTUP
TCINIT
TEPLOT	
TRANCO
TRIANG	
ZBC	

Figure 1. Cross-Reference of Subroutines
and COMMON Blocks

To find all of the COMMON blocks used by a particular subroutine, locate that subroutine in the left-hand column and read across.

To find all of the subroutines which use a particular COMMON block, locate that COMMON block in the top row and read down.

TABLE III

COMMON BLOCK CONSTRUCTION

C1: DIMENSION(25)				
1	CFEQ1	14	UO	
2	CFEQ2	15	GAM1	
3	CRES1	16	GAM2	
4	CRES2	17	PI	
5	CK11	18	TCRO	
6	CK12	19	BLANK	
7	CKE2	20	BLANK	
8	ITCVER	21	BLANK	
9	CLNLM2	22	BLANK	
10	CKIP1	23	BLANK	
11	CKEP1	24	BLANK	
12	EMASS	25	BLANK	
13	PMASS			
C2: DIMENSION(5)				
1	JCHK(1)	4	JCHK(4)	
2	JCHK(2)	5	JCHK(5)	
3	JCHK(3)			
C3: DIMENSION(200+VAR(X))				
1-175	DA(175)			
176-200	DADT(25)			
201- X	VAR(X)			
		X(MAX)=1400		
C4: DIMENSION(55)				
1- 10	IBCRR(10)	41- 50	ISHK(10)	
11- 20	IBCR(10)	51- 53	NZP(3)	
21- 30	IBCZZ(10)	54	IMPSHK	
31- 40	IBCZ(10)	55	JSTEP	
C5: DIMENSION(162)				
1-162	G(9,9,2)			
C6: DIMENSION(31)				
1	ISTEP	12- 21	IVV(10)	
2- 11	IV(10)	22- 31	IAV(10)	
C7: DIMENSION(320)				
1-100	A(10,10)	301-310	V(10)	
101-200	B(10,10)	311-320	COL(10)	
201-300	C(10,10)			
C8: DIMENSION(196)				
1- 4	TCZM(4)	29- 68	DTCDVM(4,10)	
5- 8	TCRM(4)	69-108	DTCDV(4,10)	
9- 12	TC(4)	109-148	DTCDVP(4,10)	
13- 16	TCRP(4)	149-188	DTDVDX(4,40)	
17- 20	TCZP(4)	189-192	W(4)	
21- 24	DTCDR(4)	193-196	WW(4)	
25- 28	DTCDZ(4)			

C9: DIMENSION(90)

1- 10 VARIZM(10)
11- 20 VARIRM(10)
21- 30 VARI(10)
31- 40 VARIZP(10)
41- 51 VARIZP(10)

51- 60 DVDR(10)
61- 70 DVDZ(10)
71- 80 D2VDR2(10)
81- 90 D2VDZ2(10)

C10: DIMENSION(16)

1 RM
2 R
3 RP
4 DRM
5 DRP
6 DRMP
7 DRPM
8 DDRPM

9 ZM
10 Z
11 ZP
12 DZM
13 DZP
14 DZMP
15 DZPM
16 DDZPM

C11: DIMENSION(56)

1- 4 BEZM(4)
5- 8 BERM(4)
9- 12 BE(4)
13- 16 BERP(4)
17- 20 BEZP(4)
21- 24 BEPZM(4)
25- 28 BEPRM(4)

29- 32 BEP(4)
33- 36 BEPRP(4)
37- 40 BEPZP(4)
41- 44 DBEDR(4)
45- 48 DBEDZ(4)
49- 52 DBEPDR(4)
53- 56 DBEPDZ(4)

C13: DIMENSION(10)

1- 10 IDDX(10)

C14: DIMENSION(3)

1 BZEINT
2 BZEMAX
3 BZEF

C15: DIMENSION(20)

1- 4 TCZDM(4)
5- 8 TCDRM(4)
9- 12 TCD(4)

13- 16 TCDRP(4)
17- 20 TCDZP(4)

C20: DIMENSION(7)

1 NDA
2 NRZ
3 NTV
4 NAT

5 NAU
6 NAF
7 NAE

C21: DIMENSION(150)

1-150 RZ(150)

C22: DIMENSION(20)

1- 10 JDDR(10)
11- 20 JDDZ(10)

C23: DIMENSION(100)

1 IFLAG
2 NCOM23
3- 8 KKRR(2,3)
9- 14 DDZZ(2,3)
15- 24 KKRZ(2,5)
25- 34 CKRZ(2,5)
35- 39 BSQT(5)
40- 54 BSQT(3,5)
55- 58 CBSQT(4)

59- 73 CBSQ(3,5)
74- 75 DKRRDR(2)
76- 77 DKZZDZ(2)
78- 79 DKRZDR(2)
80- 81 DKPZDZ(2)
82- 83 DTDXP(2)
84- 85 DTDXM(2)
86 DRODXP
87 DRODXM
88-100 BLANK

C25: DIMENSION(40)

1- 10 CVRMZP(10)
11- 20 CVRPZP(10)

21- 30 CVRMZM(10)
31- 40 CVRPZM(10)

C26: DIMENSION(16)

1- 4 BERMZP(4)
5- 8 BERPZP(4)

9- 12 BERMZM(4)
13- 16 BERPZM(4)

C27: DIMENSION(32)

1- 4 TCRMZP(4)
5- 8 TCRPZP(4)
9- 12 TCRMZM(4)
13- 16 TCRPZM(4)

17- 20 TDRMZP(4)
21- 24 TDRPZP(4)
25- 28 TDRMZM(4)
29- 32 TDRPZM(4)

C30: DIMENSION(5)

1 INP
2 IOUT1
3 IOUT2

4 INP2
5 IOUT4

TABLE IV

EQUIVALENCE REFERENCE OF IMPORTANT VARIABLES

DA(1)	..	NDIM	DA(66)	..	RMAX
DA(2)	..	IVERS	DA(67)	..	INTCON
DA(3)	..	MAT	DA(68)	..	JBCZZ(1)
DA(4)	..	ITC	DA(69)	..	JBCZZ(2)
DA(5)	..	IDATE	DA(70)	..	BLANK
DA(6)	..	JBCR(1)	DA(71)	..	IDA(1)
DA(7)	..	JBCR(2)	DA(72)	..	IDA(1)
DA(8)	..	NDT	.	.	.
DA(9)	..	NDR	DA(149)	..	IDA(40)
DA(10)	..	NDZ	DA(150)	..	IDA(40)
DA(11)	..	DT	DA(151)	..	BLANK
DA(12)	..	DRMIN	DA(152)	..	BLANK
DA(13)	..	DZMIN	DA(153)	..	BLANK
DA(14)	..	IPIVOT	DA(154)	..	BLANK
DA(15)	..	ISTEPO	DA(155)	..	NDIMO
DA(16)	..	NV	DA(156)	..	JJDDR
DA(17)	..	IFROM	DA(157)	..	JJDDZ
DA(18)	..	STRTYM	DA(158)	..	NTCDRV
DA(19)	..	IFYL	DA(159)	..	LASTDT
DA(20)	..	NAODA	DA(160)	..	BLANK
DA(21)	..	IIV	DA(161)	..	BLANK
DA(22)	..	IIVV	DA(162)	..	BLANK
DA(23)	..	ITAPE	DA(163)	..	BLANK
DA(24)	..	IBE	DA(164)	..	ICCOORD
DA(25)	..	RMIN	DA(165)	..	ISO
DA(26)	..	ADDA(1)	DA(166)	..	JBCRR(1)
DA(27)	..	ADDA(2)	DA(167)	..	JBCRR(2)
.	.	.	DA(168)	..	JBCR(1)
DA(51)	..	ADDA(26)	DA(169)	..	JBCR(2)
DA(52)	..	ADDA(27)	DA(170)	..	JJDDZ(1)
		MATALT	DA(171)	..	JJDDZ(2)
DA(53)		ADDA(28)	DA(172)	..	ISHOCK(1)
		IIAV	DA(173)	..	ISHOCK(2)
DA(54)	..	ADDA(29)	DA(174)	..	BLANK
		IPLAVC	DA(175)	..	BLANK
DA(55)	..	ADDA(30)			
		IDVDX			
DA(56)	..	ADDA(31)	DADT(1)	..	TIME
		NREG	DADT(2)	..	NNDT
DA(57)	..	ADDA(32)	DADT(3)	..	ISTPNO
		ZZ	DADT(4)	..	INEG
DA(58)	..	ADDA(33)	DADT(5)	..	MMAT
		NDZ2	DADT(6)	..	IITC
DA(59)	..	ADDA(34)	DADT(7)	..	JSTPNO
		RZ	DADT(8)	..	BLANK
DA(60)	..	ADDA(35)	DADT(9)	..	BLANK
		NDR2	.	.	.
DA(61)	..	ADDA(36)	.	.	.
		Z1	.	.	.
DA(66)	..	ADDA(37)	DADT(25)	..	BLANK
		NDZ1			
DA(63)	..	ADDA(38)			
		R1			
DA(64)	..	ADDA(39)			
		NDR1			
DA(65)	..	ADDA(40)			
		ZMAX			

APPENDIX G
ARRAY DESCRIPTIONS

DA ARRAY: This array is dimensioned to 175 in COMMON/C3/ and forms a part of file RECORD. It contains control constants, initial values, and data labels. Its data is used to start, restart, and output the program. It contains, as subsets, the arrays IDA and ADDA which are described below. Refer to table IV of appendix F for a detailed breakdown of this array.

DADT ARRAY: This array is dimensioned to 25 in COMMON/C3/ and forms a part of file RECORD. It contains control parameters which change with every time step. Its data is used to keep track of time parameters in the execution of the program. This array, together with the VAR array, is buffered out to external storage on each time step which is saved. Refer to table IV of appendix F for a detailed breakdown of this array.

VAR ARRAY: This array is dimensioned in COMMON/C3/ and forms a part of file RECORD. Its dimension must be determined before compilation of the program since IBM FORTRAN does not allow for variable dimensioning of arrays. This dimension is determined from:

$$\begin{aligned} \text{NVAR} &= \text{NV} * \text{NDR} (\text{NV} + \text{NDZ} + 3) && \text{if NDR is greater than NDZ} \\ \text{NVAR} &= \text{NV} * \text{NDZ} (\text{NV} + \text{NDR} + 3) && \text{if NDZ is greater than NDR} \end{aligned}$$

where NVAR is the dimensionality of the VAR array, NV is

the number of variables to be calculated, and NDR and NDZ are the number of mesh points in the 'R' and 'Z' directions respectively. After NVAR is determined, this value must be entered in the VAR declaration of COMMON/C3/ in all the subroutines which use COMMON/C3/ and in the NVAR assignment statement near the beginning of the main program. This must be done before compilation of the program. Refer to figure 1 of appendix F to find all the subroutines which use COMMON/C3/. The VAR array contains the calculated values of all the variables at all of the mesh points for a specific time step. Together with the DADT array, VAR is buffered out to external storage on each time step that is to be saved.

RZ ARRAY: This array is dimensioned to 150 in COMMON/C21/ and forms a part of file RECORD. Actually only a portion of this array is used, a block equal to NDR+NDZ. The values in this array are used to locate a given point in the R-Z plane. The first NDR words give the radius, and the last NDZ words give the 'Z' distance. Thus a mesh point (i,j) is located at $r_i = RZ(i)$ and $z_j = RZ(NDR+j)$. This array is buffered out to external storage after the DA array.

ADDA ARRAY: This array is dimensioned to 40 in a DIMENSION statement and is then equivalenced to DA(26). It contains the values of miscellaneous additional data words. Most of

these values are those of initial data read into the initializing subroutines. This array is dimensioned to 40, but the number of words should be held to 26 or less because other variables are equivalenced to the DA array beginning at DA(52) (see table of appendix F). If a value were assigned to ADDA(27) by some subroutine, then the value of MATALT would also change and could cause an undesired effect in the execution of the program. The ADDA array values are assigned in subroutines MATRIX, TCINIT, BEXINT, MESH, and INIT4. This array together with the array IDA are buffered out to the line printer by subroutine RUNDAT.

IDA ARRAY: This array is dimensioned to 40 in a DIMENSION statement and is then equivalenced to DA(71). It contains the Holerith description of the values contained in the ADDA array. The CDC FORTRAN allows a Holerith assignment of eight characters to an integer variable. Thus a direct equivalence to the DA array could be made in that language. However, the IBM FORTRAN does not allow direct Holerith assignment to a variable. Such an assignment must be made through a DATA statement. In addition, the IBM machine will only allow four characters to be assigned to a REAL*4 variable. This necessitated declaring the IDA array to be REAL*8 and expanding the DA array by forty words to accommodate the equivalenced IDA array. Since the RUNDAT subroutine outputs the IDA array instead of

the DA array, no additional indexing was necessary to handle the two-word strings. The IDA array Holerith strings are assigned in subroutines MATRIX, TCINIT, BEXINT, MESH, and INIT⁴. This array together with the array ADDA is buffered out to the line printer by subroutine RUNDAT. NADDA is the total number to be printed.

A, B, C, and V ARRAYS: The A, B, and C arrays are 10 by 10 matrices and the V array is a ten component vector. These arrays are declared in COMMON/C7/ and are used in the difference equations 3.2-1, 3.5-1, and 3.5-2 of UCRL 51103.

E and F ARRAYS: These are ten component vectors which are declared in a DIMENSION STATEMENT and then equivalenced to the VAR array. They are used in forward-backward (or backward-forward) passes to solve the difference equations in sections 3.2 and 3.5 of UCRL 51103.

W ARRAY: The W array is a reference vector which contains all the variables, excluding alternate variables, which can be calculated by this program. Alternate variables are derived from selected W vector components by multiplying or dividing by the density. The vector W is defined by $W=(RO,VR,VP,VZ,TI,TE,BR,BP,BZ,SI)$.

U ARRAY: The U array is a rearrangement of the W vector. The order of the rearrangement is determined by the IV, IIV, and IAV transformation vectors which in turn are

determined by JAV and JVERS in subroutine MATRIX. After U is determined, only the first NV components are calculated, NV also being automatically determined in subroutine MATRIX. JAV and JVERS are parameters on input card number 7.

TEMP ARRAY: The TEMP array is a temporary vector identical to the U vector. It is used during calculations and contains the values of the U vector along a single line adjacent to the U vector line at the new time step. For example, during calculations the U array contains U^{n+1} for line 'k'. At that time, the TEMP array would contain U^{n+1} for line 'k-1'.

IV, IVV, and IAV ARRAYS: These arrays are determined automatically by JAV and JVERS in subroutine MATRIX. The arrays are assigned values from integer constants IIV, IIVV, and IIAV which are assigned in the subroutine. IV(1) is the value of the first digit of IIV, IV(2) the value of the second, and so forth. The IV array determines the arrangement of the U vector. This is best explained by an example. If IV(6)=4, then the sixth component of W (TE) is placed as the fourth component of U. Again if IV(3)=6, then the third component of W becomes the sixth component of U. NV determines the number of components which will actually be calculated. Only the first NV components of U are calculated. Therefore, if IV(K) is

greater than NV, then the Kth component will not be calculated.

The IVV array essentially gives the same information that the IV array does, only in reverse. For example, if $IV(6)=4$, then $IIV(4)$ would equal six and would indicate that the fourth component of U is the sixth component of the reference vector W.

The IAV array operates in a similar manner and is concerned with alternate variables. Here $IAV(6)=1$ says that the first alternate variable for the sixth component of W is to be calculated. In this case PE (electron pressure) is calculated instead of TE (electron temperature). One further example should put these transformations in perspective.

```

W=(RO,VR,VP,VZ,VI,TE,BR,BP,BZ,SI)
IIV= 1 2 6 7 3 4 8 9 5 0      with NV=5
IIVV= 1 2 5 6 9 3 4 7 8 0
U=(RO,VR,VI,TE,BZ,VP,VZ,BR,BP,SI)
    'not calculated since NV=5

```

These arrays are declared and passed to other subroutines in COMMON/C6/.

Use of these transformation arrays permits efficient utilization of storage and time. The code is written to calculate all of the variables in W. If not all of these variables are required, much time and storage area would be wasted in their calculation. These arrays allow the operator to select the variables to be calculated and suppress all the others. Several subsets of W are already

set up in subroutine MATRIX. Any additional subsets which may be required would have to be added to MATRIX. The procedure for adding subsets is outlined in the description of subroutine MATRIX in appendix D.

IBCRR and IBCR ARRAYS: These arrays are dimensioned and declared in COMMON/C4/ and determine the inner and outer 'R' boundary conditions respectively. The array values are computed from the values of integer constants JBCRR and JBCR which are read from input card number 5A. IBCRR(1) is the value of the first digit of JBCRR, IBCRR(2) is the value of the second digit, and so forth. The same procedure determines the values for the IBCR array. The following example demonstrates the use of these arrays in determining the boundary conditions. If JBCRR=3127463100, then IBCRR(1)=3, IBCRR(2)=1, IBCRR(5)=4, and so forth. IBCRR(1)=3 indicates that the first component of W (RO) has inner radial boundary condition number three. Again IBCRR(5)=4 indicated that the fifth component of W (TI) has inner radial boundary condition number four. Boundary conditions are discussed under the subroutine BCR in appendix D.

IBCZZ and IBCZ ARRAYS: These arrays are dimensioned and declared in COMMON/C4/ and determine the lower and upper 'Z' boundary conditions respectively. The array values are computed from the values of integer constants JBCZZ

and JBCZ which are read from input card number 5A. The array values are determined and utilized in the same manner as the IBCRR and IBCR array values are used. Boundary conditions are discussed under the subroutine BCZ description in appendix D.

JDDR and JDDZ ARRAYS: These arrays are declared and dimensioned in COMMON/C22/ and determine the use of one-sided derivatives (page 79 of UCRL 51103). The array values are computed from the values of integer constants JJDDR and JJDDZ which are read from input card number 6. This computation scheme is the same one used for IBCRR and IBCR above. The array elements can only take on the values '1' or '0'. Zero indicates no one-sided derivative and one indicates use of the one-sided derivative. If JJDDR(5)=1, the fifth component of W (TI) will be calculated using one-sided derivatives.

NOTE: Throughout this paper the terms array and vector are sometimes used interchangeably. This arises from the changes Lindemuth made to conserve storage area. The variable arrays were rewritten as singly subscripted variables and thus became vectors instead of arrays. In the context of this paper the vector may be considered to be a one-dimensional array.

APPENDIX H

DATA SETS

A data set is a collection of information which resides on a particular device. Typical data sets are the card reader, card punch, line printer, plotter, magnetic disc, magnetic tape, data cells, and paper tape. The magnetic core in the CPU is also a data set. It is an internal data set and is not treated like the external data sets mentioned above. Data sets must be declared and defined in the JCL program.

The data set is declared and defined in the data definition (DD) statement. The DD statement is composed of three sections which are (from left to right) the identification field, the operation field, and the operand field.

A. IDENTIFICATION FIELD

In this field the JCL program stepname and data set identifier are specified. The following identification field

```
//GO.FT10F001
```

identifies a data set FT10F001 which is used in the GO step of a program or catalogued procedure. The third and fourth digits of this identifier form the integer device identifier which is used in the READ/WRITE statements in the source program. For example READ(10,50) A would cause variable 'A' to be transferred from core to the device

which contains data set FT10F001.

The computer facility at the Naval Postgraduate School has specified that data sets FT05F001, FT06F001, and FT07F001 reside on the card reader, line printer, and card punch respectively. These data sets do not have to be defined in the JCL program. If these devices were to be used with different data set identifiers, defining DD statements would have to be included in the JCL program.

Data sets FT06F001 and FT07F001 are used in the GO step of the common catalogued procedures. If the card punch or line printer is to be used during the FORT step (compilation of FORTRAN) the data sets would be SYSPRINT and SYSPUNCH respectively. These data sets do not have to be declared by the operator. Frequently SPACE parameters in these data sets must be over-ridden when a very large listing or object deck is expected during compilation of a source deck. The SPACE parameter is discussed in paragraph C below and over-ride procedures in appendix B.

B. OPERATOR FIELD

The operator field in all data definition statements contains 'DD'. This identifier must be preceded and followed by at least one space.

C. OPERAND FIELD

This field contains all the information necessary to completely describe the data set. The following parameters

follow the DD entry and may appear in any order. However, they must be separated by a comma and no space. Spaces are not permitted to appear anywhere within the operand field except after the end of the field or on a continuation card between the comma following the last parameter and the 'C' in column 72. A space anywhere else will terminate the program.

1. SPACE Parameter

This parameter specifies the amount of storage to be allocated for the data set. Space is allocated by TRACKS (TRK) or CYLINDERS (CYL). A track can store 7294 bytes of information. A cylinder consists of 20 tracks. For general information, a 2314 disc contains 200 cylinders.

The space parameter size is determined from the data storage requirement and the BLOCKSIZE specified in the DCB parameter discussed below. First determine the total amount of storage in bytes required to hold the expected output data. Divide this figure by the blocksize to determine the total number of blocks in the data set. Next divide this figure by the largest number of blocks which will fit in a track. This will give the number of tracks required. If allocation is to be made in cylinders, divide the number of tracks by 20 and add one for any fractional part of a cylinder.

Allocation in tracks allows the machine more flexibility in locating space since the tracks do not have to be consecutive. However, there is advantage to allocating

space in cylinders. If it is anticipated that a large portion of the data set will be used at one time, allocation should be in cylinders because then the mechanical read/write arm does not have to move during operation on a particular cylinder. If the allocation were in tracks, it is conceivable that the arm would have to move after the operation on each track was completed. This could be very time consuming if many I/O commands were made to that data set.

The SPACE assignment will take the form

```
SPACE=(TRK,(15,2),RLSE)
```

which will allocate 15 tracks initially. If the 15 tracks are insufficient, additional tracks will be allocated two at a time until 18 additional tracks have been added. RLSE will release any unused area after the program has terminated. The RLSE parameter is normally used only if the required size of the data set is unknown and a very high estimate is used to assure enough space. Thus the SPACE assignment frequently appears in the form

```
SPACE=(CYL,(3,1))
```

where three cylinders are initially allocated with increments of one being allocated as needed. Care must be taken that all commas and parenthesis appear in the proper location. This has been a source of numerous errors on the part of the author during this conversion.

2. UNIT Parameter

The UNIT parameter specifies the device upon which the data set will reside. There are identifiers for each device in use with the CPU. This paper will be concerned only with magnetic discs for data sets other than the card reader, line printer, and card punch.

For permanent data sets, the UNIT parameter will always be 2314 which is the identifier for a magnetic disc device which is used by general users. There are presently three discs available for the general user. The particular disc on which the data set is to reside is specified by VOL=SER='name' where 'name' is either MARY, LINDA, or DUFFY. The operator should check with the dispatcher to assure that adequate space is available for a particular disc before completing this parameter. The UNIT assignment will then take the form

UNIT=2314,VOL=SER=MARY

which places the data set on a 2314 disc pack named MARY.

For temporary data sets, the assignment is UNIT=SYSDA. SYSDA is reserved for general user temporary data sets. The data sets are destroyed after the program execution is terminated. When UNIT=SYSDA is used, VOL=SER='name' and DISP, which is discussed below, are not required. This unit assignment was used for all of the data sets during the testing phase of the conversion when it was not important that the information be saved.

3. Data Control Block (DCB) Parameter

The DCB parameter describes the construction of the data set. It is composed of three parts; the record format, the logical record size, and the block size.

a. Record Format

The record format (RECFM) specifies the type of grouping of data elements in the data set. Typical RECFM's are fixed blocked, variable blocked, variable blocked spanned, and so forth. This program uses fixed blocking and the assignment if RECFM=FB.

b. Logical Record Length (LRECL)

The logical record size is the length in bytes of the list in the READ/WRITE statement used for the data set. For example, the statement

```
WRITE(10) A,B,C
```

where A, B, and C are REAL*4 variables would require a logical record length of 12 since the list contains three words at four bytes per word. There are eight bits per byte.

In this program, file RECORD is placed in data set FT10F001. The LRECL for this data set is 100. Thus to read or write vectors of greater length than 25 words, the I/O commands were rewritten in DO loops so that each READ/WRITE statement has a list of only 25 words in length.

c. Block Size (BLKSIZE) Parameter

The block size parameter is specified to be an integer multiple of the LRECL. The normal practice is

to dimension the BLKSIZE such that two blocks will fit on one track. Thus the maximum BLKSIZE will be 3520. To determine the block size for the data set, divide 3520 by the LRECL. The truncated integer result multiplied by the LRECL will be the BLKSIZE in bytes. In this program the BLKSIZE is 3500.

The DCB parameter will take the form

DCB=(RECFM=FB,LRECL=100,BLKSIZE=3500)

which indicates that the data set consists of fixed blocks of 35 logical records. The logical records are 100 bytes long and there are two blocks per track of disc.

4. Disposition (DISP) Parameter

The disposition parameter defines the status and disposition of the data set and is composed of two sub-fields.

a. Status

This sub-field may contain either OLD, NEW, MOD, or SHR. OLD indicates an existing data set while NEW indicates a data set that is being created. MOD indicates an existing data set which may be modified and SHR indicates that the data set may be shared by other jobs.

b. Disposition

This parameter may be either DELETE, KEEP, or PASS. DELETE means that the storage occupied by the data set will be released when the job is terminated. KEEP says to keep the data set for later use, and PASS says to pass the data set to succeeding steps in the job.

Thus the statement `DISP=(NEW,KEEP)` means keep the data set being created after the job is over.

5. Data Set Name (DSNAME) Parameter

DSNAME names the data set and takes the form `DSNAME=Unnnn.anyname` where 'U' indicates the type of user, nnnn is the user's assigned number, and 'anyname' is a unique name assigned by the user to identify the data set. This name is limited to six alphanumeric characters. The type of user will be 'S' for student, 'F' for faculty, 'C' for computer center staff, 'N' for NPS staff, and 'X' for external users. The statement `DSNAME=S0833.RECORD` gives the name RECORD to a data set used by a student whose user number is 0833. The DSNAME parameter is not required when defining temporary data sets on unit SYSDA.

6. Label (LABEL) Parameter

The label parameter is used to indicate the expiration date of a permanent data set and may take two forms:

`LABEL=EXPDT=yyddd` where yy is the year and ddd is the Julian day of the year.

`LABEL=RETDP=zzz` where zzz is the number of days from its creation that the data set is to be retained.

The statement `LABEL=EXPDT=72366` indicates that the data set will be retained until 31 Dec 72.

A complete DD description will now take the form

```
//GO.FT10F001 DD UNIT=2314,DSNAME=S0833.RECORD      C
//  DISP=(NEW,KEEP),VOL=SER=MARY,                    C
//  LABEL=EXPDT=72366,SPACE=(TRK,(20,2)),             C
//  DCB=(RECFM=FB,LRECL=100,BLKSIZE=3500)
```


The source program statement WRITE(20) A will transfer the 32 bit pattern under the identifier 'A' from core into the data set named RECORD. This data set is located on disc pack MARY and will be retained until 31 Dec 72.

APPENDIX I
IDENTIFIER DESCRIPTIONS

This section contains brief descriptions of the initial value and control identifiers. The following is not a complete list of all the identifiers used in the program, but rather a list of the identifiers which set initial values and control the execution of the program through subroutine argument lists and COMMON blocks.

BP	Component of B field in the phi direction
BPF	Frequency plasma B field in the phi direction
BPINIT	Initial plasma B field in the phi direction
BPMAX	Maximum plasma B field in the phi direction
BR	Component of B field in the R direction
BRAT	Mirror ratio
BZ	Component of B field in the Z direction
BZF	Frequency of plasma B field in the Z direction
BZEF	Frequency of external B field in the Z direction
BZEINT	Initial external B field in the Z direction
BZEMAX	Maximum external B field in the Z direction
BZINIT	Initial plasma B field in the phi direction
BZMAX	Maximum plasma B field in the Z direction
CLNLM2	Constant used in modified Coulombian logarithm
CQA	Shock smoothing used
DT	Size of time step
ECHG	Electron charge

EMASS	Electron mass
EO	Permittivity
GAM	Gamma Factor
IARRAY	Selects array to be buffered in or out by BUFFER
IBE	IBE=1 separates B field into plasma and external components and reads in external field components
IDATEO	Julian data, five character
IFROM	Selects restart mode. IFROM ≠ 0 restarts from file OLDREC. IFROM=0 restarts from file RECORD.
IFYL	File reference number
INP	Data set FT05F001 identifier
INP2	Data set FT30F001 identifier
INTCON	Selects initial condition subroutines - '1' selects INIT1, '4' selects INIT4, and '5' selects INIT5.
IOP	Selects profile option
IOUT1	Data set FT10F001 identifier
IOUT2	Data set FT06F001 identifier
IOUT4	Data set FT40F001 identifier
IPU	Selects restart mode - '0' allows new parameters to be read in MATRIX on restart. '1' causes code to use original parameters
IROR	Determines read or write function in BUFFER
ISHOCK	Selects variables to receive shock smoothing
ISO	Selects isotropic or anisotropic thermal conductivities
IST	Gives the number of the time step from which restart is to begin
ISTEP	Selects coding for different passes through equations
ISTEPO	Used to set ISTEP odd or even

ITAPE	Tape reference number
ITCVER	Subroutine version
ITP	File identifier in READ/WRITE and BUFFER commands
JAV	Selects alternate variables to be calculated
JBCR	Sets outer R boundary conditions
JBCRR	Sets inner R boundary conditions
JBCZ	Sets upper Z boundary conditions
JBCZZ	Sets lower Z boundary conditions
JDDR	Selects one-sided derivatives for R
JDDZ	Selects one-sided derivatives for Z
JVERS	Selects variables to be calculated
J2BCHK	Gives R coordinate of mesh point to be checked
K2BCHK	Gives Z coordinate of mesh point to be checked
MATALT	Selects coding subset in MAT2. This code uses MATALT=0.
MMAT	Selects difference method
NDIM	Number of dimensions
NDR	Number of mesh points in the R direction
NDT	Number of time steps
NDTBUF	Arrays DADT and VAR will be saved every NDTBUF time step
NDTPNT	OUTPT1 will be called every NDTPNT time steps
NDZ	Number of mesh points in the Z direction
NREG	If greater than '1', card 11B is read
NTCDRV	NTCDRV=1 indicates no transport coefficient derivatives
NZP(K)	Number of the Kth Z line to be printed by OUTPT1

N2BCHK	Gives number of mesh points which will be checked by CHECK and SPECHK
PMASS	Mass of the proton or ion
PI	
RMAX	Maximum radius
RMIN	Minimum radius
RO	Density
ROINIT	Peak density
ROMIN	minimum density
RRAT	Determines mesh spacing factor in the R direction
SI	PSI, the stream function
SMASS	EMASS+PMASS
STKEQO	STKEQO=1 sets thermal conductivity equal to zero
SV	Specific volume
TE	Electron temperature
TI	Ion temperature
TINIT	Peak temperature
TMIN	Plasma background temperature
UO	Permeability
VEQMAX	Where velocity is maximum
VEQO	Multiple of XO where velocity goes to zero
VO	Plasma initial boundary velocity
VP	Component of velocity in the phi direction
VR	Component of velocity in the R direction
VZ	Component of velocity in the Z direction
XO	Plasma initial boundary radius
XO	Plasma initial boundary radius

ZMAX	Maximum distance in the Z direction
ZRAT	Determines the mesh spacing factor in the Z direction

CONVERTED SOURCE PROGRAM 3 JUN 72

PROGRAM IRLSMHD ... MHD EQUATIONS USING ADI METHOD
 THE MAIN PROGRAM USES FILES RECORD AND MHDOUT.
 THE MAIN PROGRAM USES INPUT CARDS NUMBER 1, 2, AND 3.

DATA SET REFERENCE NUMBERS

5 DATA (CARD READER)
 6 MHDOUT (LINE PRINTER)
 10 RECORD (DISC)
 30 OLDREC (DISC)
 40 SCRAT (DISC)

MANUAL CONTROL THROUGH THE SENSE SWITCH ARRAY IS NOT AVAILABLE
 AT THIS TIME.

JCHK ARRAY OPTIONS

JCHK(1)=1 ARRAYS A,B,C,V ARE PRINTED
 JCHK(2)=1 ARRAYS E,F ARE PRINTED
 JCHK(3)=1 ARRAY I IS PRINTED
 JCHK(4)=1 CALCULATION OF E,F IS CHECKED
 JCHK(5)=1 CALCULATION OF VARIABLES AT NEW TIME STEP IS CHECKED

COMPLETE MHD VECTOR W IS U=(RO,VR,VP,VZ,TE,BR,BP,BZ,SI)

INTEGER COL,XXX

COMMON/C1/DUMMY(25)

COMMON/C2/JCHK(5)

COMMON/C3/DAT(175),DADI(25),VAR(3500)

COMMON/C4/IBCCR(10),IBCR(10),IBZZ(10),IBCZ(10),ISHK(10),NZP(3),

1 IMPSHK,JSTEP

COMMON/C5/G(9,2)

COMMON/C6/ISTEP,V(10),IVV(10),IAV(10)

COMMON/C7/A(10,10),B(10,10),C(10,10),V(10),COL(10)

COMMON/C20/NOA,NRZ,NIV,NAT,NAU,NAF,NAE

COMMON/C21/RZ(150)

COMMON/C23/DUM23(100)

COMMON/C30/INP,IOUT1,IOUT2,INP2,IOUT4

DIMENSION J2BCHK(10),K2BCHK(10)

DIMENSION BC(10,10,2)

DIMENSION U(10),TEMP(10),F(10),E(10)

10
 20
 30
 40
 50
 60
 70
 80
 90
 100
 110
 120
 130
 140
 150
 160

EQUIVALENCE	(BC(1),B(1))	170
EQUIVALENCE	(DA(1),NDIM), (DA(2), IVERS), (DA(3), MAT), (DA(4), ITC),	180
	(DA(5), IDATE), (DA(8), NOT), (DA(11), DT), (DA(12), DRMIN),	190
1	(DA(13), DZMIN), (DA(14), IPIVOT), (DA(15), ISTEP0),	200
2	(DA(16), NV), (DA(17), IFROM), (DA(18), STRTYM),	210
3	(DA(19), IFVL), (DA(20), NADDA), (DA(21), IIV),	220
4	(DA(22), IIVV), (DA(23), ITAPE), (DA(24), IIBEL),	230
5	(DA(25), RMIN), (DA(26), CQA), (DA(54), IPLAVC),	240
6	(DA(67), INICON), (DA(159), LASTDT)	250
7	(DA(155), NDIM0), (DADT(1), TIME), (DADT(2), NNDT), (DADT(3), ISTEPNO),	260
8	(DADT(4), INEG), (DADT(5), MMAT), (DADT(7), JSIPNO)	270
9	(DADT(1), U(1), TEMP(1), F(1), E(1))	280
EQUIVALENCE	(VAR(1), U(1), TEMP(1), F(1), E(1))	290
SET DATA SET		300
LABELS		
IOUT1=10		310
INP=5		320
IOUT2=6		330
INP2=30		340
IOUT4=40		350
READ(INP,1) N2BCHK		360
FORMAT(I2)		370
1 READ(INP,2) (J2BCHK(J),K2BCHK(J),J=1,N2BCHK)		380
2 FORMAT(20I2)		390
3 READ(INP,3) NDTBUF		400
FORMAT(I13)		410
IF(NDTBUF.EQ.0) NDTBUF=1		420
MOUT=1		430
NVAR=3500		440
NDA=175		450
ISU=10		460
ISU=ISU		470
IIP=IOUT1		480
IOUT=6		490
ISUB=1		500
XXX=1		510
WRITE(6,5000) XXX		520
FORMAT(1) CHECK POINT ,I1)		530
5000 CALL OVRISU(ISUB,ISU,IRET,IDUM)		540
58 XXX=2		550
WRITE(6,5000) XXX		560
IPNT=IDUM		570
NDTPNT=ISUB		580
ISAVDT=NNDT-NDTBUF*(NNDT/NDTBUF)		590
NTV=NV*NDR*NDZ		600
NRZ=NDR*NDZ		610
NX=NDR		620
IF(NDZ.GT.NDR) NX=NDZ		630

C


```

640 NAT=NTV
650 NAU=NAU+NX*NV
660 NAF=NAU+NX*NV
670 NAE=NAF+NX*NV
680 TEMP(J,K)=TEMP(NAT+J+NV*(K-1))
690 U(J,K)=U(NAU+J+NV*(K-1))
700 E(J,K)=E(NAF+J+NV*(K-1))
710 E(J,K,L)=E(NAE+J+NV*(K-1+NV*(L-1)))
720 IF((NAE+NX*NV).LE.NVAR) GO TO 59
730 WRITE(IOUT2,16,60)
740 FORMAT(' INSUFFICIENT STORAGE')
750 STOP
760 59 GO TO (125,170), IRET
770 STOP
780 125 XXX=3
790 WRITE(6,5000) XXX
800 CALL BUFFER(1,1,ITP)
810 CALL BUFFER(1,2,ITP)
820 GO TO 990
830 170 CONTINUE
840 XXX=6
850 WRITE(6,5000) XXX
860 ISTEP=3-ISTEP
870 GO TO (171,172,173,174), IVERS
880 171 ISTEP=3-ISTEP
890 IF(NDIM.EQ.1) ISTEP=1
900 GO TO 175
910 172 ISTEP=7-ISTEP
920 IF(NDIM.EQ.1) ISTEP=3
930 GO TO 175
940 173 ISTEP=ISTEP+1
950 IF(NDIM.EQ.1) AND.((ISTEP.EQ.2).OR.(ISTEP.EQ.4))) ISTEP=ISTEP+1
960 IF(ISTEP.EQ.5) ISTEP=1
970 GO TO 175
980 174 ISTEP=NDIM-4*(NNDT/4)
990 IF(ISTP.EQ.1) GO TO 171
1000 IF(ISTP.EQ.3) GO TO 171
1010 GO TO 175
1020 175 GO TO (176,177,176,177), ISTEP
1030 176 NP=NDR
1040 NN=NDZ+1-NDIM
1050 GO TO 180
1060 177 NP=NDZ
1070 NN=NDR-1
1080 180 CONTINUE
1090 IREC=0
1100 DO 910 N=NDIM,NN
1110 C BOUNDARY CONDITIONS HAVE FORM G(1)*U(1)=E(1)*U(2)+F(1) AND

```



```

C      GO TO (181,181,182,182), ISTEP
181  INN=1
     JNN=1
     JN=NP
     JN1=NP-1
     GO TO (183,184), ISTEP
182  INN=2
     JNN=2
     JN=NP
     JN1=1
     JN2=2
     ISTEP2=ISTEP-2
     GO TO (183,184), ISTEP2
     BOUNDARY CONDITIONS, ODD TIME STEP
     CALL BCZ(N)
C 183  GO TO 185
     BOUNDARY CONDITIONS, EVEN TIME STEP
     CALL BCZ(N)
C 184  CONTINUE
185  DO 305 J=1, NV
     DO 305 K=1, NV
     IF((J.NE.K).AND.(G(J,K,INN).NE.O.)) GO TO 310
305  CONTINUE
310  CONTINUE
     NFA=NAF+NV*(IN-1)
     NEA=NAE+NV*(-1+NV*(IN-1))
     DO 315 J=1, NV
     NNFA=NEFA+J
     V(J)=F(NNFA)
     NNEA=NEA+J
     DO 315 K=1, NV
     A(J,K)=G(J,K,INN)
     NNEA=NNEA+NV
     B(J,K)=E(NNEA)
     B(J,K)=E(J,K,IN)
     CALL TRIANG(O)
     DO 320 J=1, NV
     NNFA=NEFA+COL(J)
     F(NNFA)=V(J)
     F(COL(J),IN)=V(J)
     NNEA=NEA+COL(J)
     DO 320 K=1, NV
     NNEA=NNEA+NV
     E(NNEA)=B(J,K)
     E(COL(J),K,IN)=B(J,K)
C 320

```

1070
1080
1090
1100
1110
1120
1130
1140
1150
1160
1170
1180
1190
1200
1210
1220
1230
1240
1250
1260
1270
1280
1290
1300
1310
1320
1330
1340
1350
1360
1370
1380
1390
1400
1410
1420
1430
1440
1450
1460
1470
1480


```

335 CONTINUE
340 DO 340 J=1,5
    JCHK(J)=0
    IF(N2BCHK.EQ.0) GO TO 360
    DO 343 J=1,N2BCHK
        GO TO (341,342,341,342), ISTEP
    341 IF(N.EQ.K2BCHK(J)) GO TO 345
        GO TO 343
    342 IF(N.EQ.J2BCHK(J)) GO TO 345
    343 CONTINUE
        GO TO 360
    345 READ(INP,4) (JCHK(L),L=1,5)
    4   FORMAT(5I1)
    360 CONTINUE
        NPM=NPM-1
    DO 710 JJ=2,NPM
        GO TO (410,410,420,420), ISTEP
    410 JJ=JJ
        LAST=JJ-1
        GO TO (430,440), ISTEP
    420 JJ=NP+1-JJJ
        LAST=JJ+1
        ISTEP2=ISTEP-2
        GO TO (430,440), ISTEP2
    430 J=JJ
        K=N
        GO TO 450
    440 J=N
        K=JJ
    450 CONTINUE
        JCHK(1)=0
        IF(N2BCHK.EQ.0) GO TO 455
        DO 453 LL=1,N2BCHK
            IF((J.EQ.J2BCHK(LL)).AND.(K.EQ.K2BCHK(LL))) GO TO 454
        453 CONTINUE
        GO TO 455
    454 JCHK(1)=1
    455 CONTINUE
        CALL SETUP(J,K)
        CALL MAT2(J,K)
        C MATRICES GENERATES DIFFERENCE EQUATIONS IN FORM A*U+B*U(+)-C*U(-)=V
        IF(IMPSX.NE.0) CALL SHKMAT(J,K)
        IF((JCHK(1).EQ.0).AND.(JCHK(5).EQ.0)) GO TO 614
        CALL CHECK(J,K,1)
    614 CONTINUE
        DO 615 LL=1,NV
            DO 615 MM=1,NV
    615 B(LL,MM)=-B(LL,MM)

```



```

NFA=NAF+NV*(LAST-1)
NEA=NAE+NV*(-1+NV*(LAST-1))
DO 620 MM=1, NV
NNFA= NFA+MM
DO 620 LL=1, NV
V(LL)=V(LL)+BC(LL,MM,JNN)*E(NNFA)
C 620 V(LL)=V(LL)+BC(LL,MM,JNN)*E(MM,LAST)
DO 630 ML=1, NV
NNEA= NEA+ML
NNEA= NNEA+NV
DO 630 LL=1, NV
A(LL,MM)=A(LL,MM)-BC(LL,ML,JNN)*E(NNEA)
C 620 A(LL,MM)=A(LL,MM)-BC(LL,ML,JNN)*E(ML,MM,LAST)
IF(ISTEP.LE.2) GO TO 660
DO 650 LL=1, NV
DO 650 MM=1, NV
B(LL,MM)=C(LL,MM)
C 660 CONTINUE
A*E(JJ)=B. A*F(JJ)=V
CALL TRIANG(O)
NFA=NAF+NV*(JJ-1)
NEA=NAE+NV*(-1+NV*(JJ-1))
DO 680 LL=1, NV
NNFA= NFA+COL(LL)
F(NNFA)=V(LL)
NNEA= NEA+COL(LL)
DO 680 MM=1, NV
NNEA= NNEA+NV
E(NNEA)=B(LL,MM)
C 680 E(COL(LL),MM,JJ)=B(LL,MM)
CONTINUE
IF(N.LE.2) GO TO 716
IF(N.EQ.3) GO TO 717
NPTS= NDR*NDZ
GO TO (711,712,711,712), ISTEP
C 711 CONTINUE
IJ= NDR*(N-3-NDZ)
INCL=1
GO TO 713
C 712 IJ=N-2+NDR*(-1-NDZ)
INCL= NDR
CONTINUE
C 713 NTA= NAT-NV
DO 714 L=1, NP
NTA=NTA+NV
IJ=IJ+INCL
IJK=IJ

```



```

DO 714 K=1,NV
NNTA=NTA+K
IJK=IJK+NPTS
714 VARIJK)=TEMP(NNTA)
717 NTUA=-NV
DO 718 L=1,NP
NTUA=NTUA+NV
DO 718 K=1,NV
NNTA=NAT+NTUA+K
NNUA=NAU+NTUA+K
718 TEMP(NNTA)=U(NNUA)
716 CONTINUE
IF((JCHK(2).EQ.0).AND.(JCHK(4).EQ.0).AND.(JCHK(5).EQ.0)) GO TO 715
CALL CHECK(N,IDUM,2)
715 CONTINUE
NEA=NAE+NV*(-1+NV*(JN-1))
DO 720 K=1,NV
NNEA=NEA+K
DO 720 L=1,NV
NNEA=NNEA+NV
IF(((K.NE.L).AND.(G(K,L,JNN).NE.0.)),OR.(E(NNEA).NE.0.)) GO TO 730
IF(((K.NE.L).AND.(G(K,L,JNN).NE.0.)),OR.(E(K,L,JN).NE.0.)) GO TO
C 1730
720 CONTINUE
KJN=NV*(JN-1)
DO 725 K=1,NV
NNUA=NAU+KJN+K
NNEA=NAF+KJN+K
725 U(NNUA)=F(NNFA)
725 U(K,JN)=F(K,JN)
GO TO 775
730 NFA=NAF+NV*(JN-1)
DO 740 K=1,NV
NNEA=NFA+K
V(K)=F(NNFA)
V(K)=F(K,JN)
C 740 L=1,NV
A(K,L)=G(K,L,JNN)
NEA=NAE+NV*(-1+NV*(JN-1))
NFA=NAF+NV*(JN-1)
DO 742 L=1,NV
NEA=NEA+NV
NFA=NFA+L
DO 742 K=1,NV
NNEA=NFA+K
V(K)=V(K)+E(NNEA)*F(NNFA)
742 V(K)=E(K,L,JN)+V(K)
C MEA=NAE+NV*(-1+NV*(JN-1))

```



```

NEA=NAE+NV*(-1+NV*(JN-1))
DO 745 M=1,NV
  NEA=NEA+NV
  MMEA=MEA+M
DO 745 K=1,NV
  NNAE=NEA+K
  MNNAE=MMEA
DO 745 L=1,NV
  MNNAE=MNEA+NV
  A(K,L)=A(K,L)-E(NNAE)*E(MNEA)
CC745 A(K,L)=-E(K,M,JN)*E(M,L,JNI)+A(K,L)
C
CALL TRIANG(1)
NNA=NAU+NV*(JN-1)
DO 770 LL=1,NV
  NNNA=NNA+COL(LL)
  U(NNNA)=V(LL)
  U(COL(LL),JN)=V(LL)
  CONTINUE
CC770
775 DO 780 J=2,NP
  DO TO (777,777,778,778),ISTEP
  L=NP+1-J
  LAST=L+1
  GO TO 779
778 L=J
  LAST=L-1
  CONTINUE
779 NFA=NAF+NV*(L-1)
  MLU=NAU+NV*(L-1)
  NEA=NAE+NV*(-1+NV*(L-1))
  NNA=NAU+NV*(LAST-1)
DO 780 M=1,NV
  MMLU=MLU+M
  MNNA=NFA+M
  U(MMLU)=F(MNNA)
  U(M,L)=F(M,L)
  NNEA=NEA+M
DO 776 MM=1,NV
  MNNA=MNEA+NV
  NNNA=NNA+MM
  U(MMLU)=U(MMLU)+E(NNEA)*U(NNNA)
  U(M,L)=U(M,L)+U(M,MM,L)*U(MM,LAST)
CONTINUE
C
776 IF((JCHK(1).EQ.0).AND.(JCHK(3).EQ.0)) GO TO 784
780 IF((JCHK(1).EQ.1) JCHK(1)=N2BCHK+1
  CALL CHECK(N,IDUM,3)
CONTINUE
784
783 CONTINUE

```


C	790	AT THIS POINT U(J) HAS VALUES OF VARIABLES AT NEXT TIME STEP	
	795	GO TO (795,840,795,840),1STEP	3260
		CONTINUE	3270
		IF(N.NE.NN) GO TO 890	3280
		IF(INDIM.EQ.1) GO TO 897	3290
		NPIS=NDR*NDZ	3300
		JJ=NDR*(NN-2-NDZ)	3310
		NTA=NAT-NV	3320
		NUA=NAU-NV	3330
		DO 805 K=1,NV	3340
		JJ=JJ+NPTS	3350
		NNUA=NUA+K	3360
		NNTA=NTA+K	3370
		DO 805 J=1,NDR	3380
		NNUA=NNUA+NV	3390
		NNTA=NNTA+NV	3400
		JK=JJ+J	3410
		KJ=JK+NDR	3420
		VAR(KJ)=TEMP(NNTA)	3430
	805	VAR(KJ)=U(NNUA)	3440
C	805	VAR(J+NDR*(NN-2+(K-1)*NDZ))=TEMP(K,J,1)	
C		VAR(J+NDR*(NN-1+(K-1)*NDZ))=U(K,J)	
C		BOUNDARY CONDITIONS, ODD TIME STEP	
		CALL ECZOTS	
		GO TO 910	
	840	CONTINUE	3450
		IF(N.NE.NN) GO TO 890	3460
		NPIS=NDR*NDZ	3470
		JJ=NN-1+NDR*(-1-NDZ)	3480
		NTA=NAT-NV	3490
		NUA=NAU-NV	3500
		DO 855 K=1,NV	3510
		JJ=JJ+NPTS	3520
		JK=JJ	3530
		NNTA=NTA+K	3540
		NNUA=NUA+K	3550
		DO 855 J=1,NDR	3560
		NNUA=NNUA+NV	3570
		NNTA=NNTA+NV	3580
		JK=JK+NDR	3590
		KJ=JK+1	3600
		VAR(KJ)=TEMP(NNTA)	3610
	855	VAR(KJ)=U(NNUA)	3620
		VAR(NN+NDR*(J-1+(K-1)*NDZ))=TEMP(K,J,1)	3630
C	855	VAR(NN+NDR*(J-1+(K-1)*NDZ))=U(K,J)	
C		BOUNDARY CONDITIONS,EVEN TIME STEP	
C		CALL BCSETS	
		GO TO 910	3650
			3660


```

897 CONTINUE
   JK=-NDR
   NUA=NAU-NV
   DO 895 K=1,NV
     JK=JK+NDR
     NNUA=NUA+K
   DO 895 J=1,NDR
     JJK=JK+J
     NNUA=NNUA+NV
     VAR(JJK)=U(NNUA)
     VAR(J+(K-1)*NDR)=U(K,J)
C 895 GO TO 910
890 CONTINUE
910 CONTINUE
   IF(CQA.EQ.0.) GO TO 915
   IF(IIMP$HK.NE.0.) GO TO 915
   CALL SHOCK(DUM1,DUM2)
   IF(NDIM.EQ.2) CALL BCZ$HK
   CALL BCR$HK
915 CONTINUE
   NNDT=NNDT+1
990 TIME=TIME+DT
   XXX=4
   WRITE(6,5000) XXX
955 CONTINUE
   ISTEPND=ISTEP
   ISTEPNG=ISTEP
   IPNT=IPNT+1
   ISAVDT=ISAVDT+1
   NPIS=NDR*NDZ
   INEG=0
   DO 1000 J=1,NDR
     DO 1000 K=1,NDZ
       JK=J+NDR*(K-1-NDZ)
       JK1=JK+IV(1)*NPIS
       JK5=JK+IV(5)*NPIS
       JK6=JK+IV(6)*NPIS
       IF((VAR(JK1).GT.0.).AND.(VAR(JK5).GT.0.).AND.(VAR(JK6).GT.0.)) GO
         1 TO 1000
       INEG=1
     GO TO 1001
1000 CONTINUE
   XXX=5
   WRITE(6,5000) XXX
   IF(1SAVDT.LT.NDTBUF) GO TO 170
   ISAVDT=0
   WRITE(ITP) DADT
1001 CALL BUFFER(1,3,ITP)

```



```

IF(INEG.EQ.0) GO TO 1002
MOUT=1
CALL OVRISU(IQUT,ISU1,MOUT,IDUM1)
WRITE(OUTPUT,1005) TIME,NDT
1005 FORMAT(1, RUN TERMINATED AT TIME=,E15.6, AFTER',15,' TIME STEPS.
1 TEMPERATURES OR DENSITY ZERO OR NEGATIVE.')
```

```

GO TO 1020
1002 CONTINUE
1010 IF(NDT.LT.NDT) GO TO 1015
CALL OVRISU(IQUT,ISU1,MOUT,IDUM1)
GO TO 1020
1015 IF(IPNT.LT.NDTPNT) GO TO 170
IPNT=0
1017 CONTINUE
CALL OVRISU(IQUT,ISU1,MOUT,IDUM1)
GO TO 170
1020 CONTINUE
1025 ENDFILE ITP
REKIND ITP
LASTDT=NDT
NDIM=NDIM
WRITE(ITP) (DA(J),J=1, NDA)
1055 CONTINUE
1060 STOP
END
```

```

SUBROUTINE BCR(K)
COMMON/C1/DUMDUM(14),UO,GAM1,GAM2,PI
COMMON/C3/ DA(175),DADT(25),VAR(3500)
COMMON/C4/ IBCRR(10),IBCR(10),IBCZZ(10),IBCZ(10),ISHK(10),NZP(3),
1 IMPSHK,JSTEP
COMMON/C5/G(9,9,2)
COMMON/C6/ISTEP,IV(10),IVV(10),IAV(10)
COMMON/C7/ALU(10),B(10,10),D(10,10),V(10),COL(10)
COMMON/C8/ICZM(4),ICRM(4),IC(4),ICRP(4),TCZP(4),DTCDR(4),DTCZ(4),
1 DTCOVN(4,10),DTCOV(4,10),DTCVDP(4,10),DTCVDX(4,10),W(4),
2 WW(4)
COMMON/C9/VARI(10),VARIRM(10),VARI(10),VARIRP(10),VARIZP(10),
1 DVDR(10),DVDZ(10),D2VDR2(10),D2VDRZ(10)
COMMON/C20/NDA,NRZ,NTV,NAT,NAU,NAF,NAE
COMMON/C21/RZ(150)
1 DIMENSION E(10),F(10)
2 DIMENSION RR(10)
DIMENSION VVAR(10),VARIM(10),VVARIP(10)
DIMENSION (DA(1),NDIM),(DA(9),NDR),(DA(10),NDZ),(DA(11),DT),
EQUIVALENCE
(DA(16),NV),(DA(25),RMIN),(DA(29),ROMIN),
(DA(30),TRO,XRO),(DA(31),BZINIT),(DA(32),BZMAX,EPMAX),
1 2
```



```

3      (DA(33),BZE,EPF),(DA(34),BRAT),(DA(35),BPINIT),
4      (DA(36),BPMAX),(DA(37),BBP),(DA(54),PLAVC),
5      (DA(65),ZMAX),(DA(66),RMAX)
EQUIVALENCE
EQUIVALENCE
EQUIVALENCE
EQUIVALENCE
1      (VAR,E,F),(RR,RZ)
2      (VAR(1),RO),(VAR(2),VR),(VAR(3),VP),(VAR(4),VZ),
3      (VAR(5),TI),(VAR(6),TE),(VAR(7),BR),(VAR(8),BP),
4      (VAR(9),BZ),(VAR(10),SI)
EQUIVALENCE
1      (VAR(1),RROR,VARIZP(1)),(VAR(2),VVR),(VVAR(3),VVP)
2      (VAR(4),VVZ),(VAR(5),TTI),(VVAR(6),TTE),
3      (VAR(7),BBR),(VAR(8),BBP),(VVAR(9),BBZ),
4      (VAR(10),SSI)
EQUIVALENCE
1      (VAR(1),ROM,VARIM(1)),(VARIM(2),VRM),(VARIM(3),VPM)
2      (VARIM(4),VZM),(VARIM(5),TIM),(VARIM(6),TEM),
3      (VARIM(7),BRM),(VARIM(8),BPM),(VARIM(9),BZM),
4      (VARIM(10),SIM)
EQUIVALENCE
1      (VARIP(1),RROR,VARIRP(1)),(VARIP(2),VVRP)
2      (VARIP(3),VVPP),(VARIP(4),VVZP),(VARIP(5),TTIP),
3      (VARIP(6),TTEP),(VARIP(7),BBRP),(VARIP(8),BBPP),
4      (VARIP(9),BBZP),(VARIP(10),SSIP)
IVZ=IV(2)
DO 10 J=1,NV
JJ=JV(J)
VARIM(JJ)=VAR(NDR*(K+(J-1)*NDZ))
VVAR(JJ)=VAR(1+(K-1+(J-1)*NDZ)*NDR)
VVARIP(JJ)=VAR(2+(K-1+(J-1)*NDZ)*NDR)
F(NAF+J)=0.
JNDR=NAF+J+NV*(NDR-1)
F(J,NDR)=0.
DO 10 L=1,NV
JL=NAE+J+NV*(L-1)
JLNDR=NAE+J+NV*(L-1+NV*(NDR-1))
E(J,L)=0.
E(J,L,NDR)=0.
DO 10 M=1,2
G(J,L,M)=0.
IF(J,EQ,L) G(J,L,M)=1.
10  CONTINUE
NVP=NV+1
DO 20 J=NVP,10
JJ=JV(J)
VARIM(JJ)=0.
VVARIP(JJ)=0.
VARI(JJ)=0.
20  VARI(JJ)=0.
DR=RR(NDR)-RR(NDR-1)

```

4600
4610
4620
4630
4640
4650
4660
4670
4680
4690
4700
4710
4720
4730
4740
4750
4760
4770
4780
4790
4800
4810
4820
4830
4840
4850
4860
4870
4880
4890
4900
4910
4920
4930
4940
4950
4960
4970
4980
4990
5000
5010
5020
5030
5040
5050
5060
5070


```

DRM=RR(2)-RR(1)
IF(IAV(1).EQ.1) RO=1./RO
DO 2000 J=1,NV
JJ1=NAE+J+NV*(J-1)
JJ2=NAF+J
JJNDR=NAE+J+NV*(J-1+NV*(NDR-1))
JJNDR=NAF+J+NV*(NDR-1)
JJ=IVV(J)
IBCCRJ=IBCCR(JJ)
IBCCRJ=IBCCR(JJ)
GO TO (100,200,300,400,500,600,700,800,900,1000),JJ
100 GO TO (101,102,103,104),IBCCRJ
101 E(J,J)=1.
102 F(J,1)=ROMIN+(RR0-ROMIN)*EXP (-DT/TR0)
103 F(J,1)=ROMIN+(RR0-ROMIN)*EXP (-VVR*DT/(XRO*DRM))
104 F(J,1)=RR0
150 GO TO (151,152,153,154,155),IBCCRJ
151 F(J,NDR)=ROMIN+(RO-ROMIN)*EXP (-DT/TR0)
IF(IAV(1).EQ.1) F(J,NDR)=1./F(J,NDR)
GO TO 2000
152 F(J,NDR)=ROMIN+(RO-ROMIN)*EXP (VR*DT/(XRO*DR))
153 E(J,J,NDR)=1.
C FROM 1D CONTINUITY EQUATION WITH RO(NDR+1)=0.=VR(NDR+1)
154 G(J,J,2)=1./DT-VR*0.5/DR+VR/RMAX
G(J,1V2,2)=0.5*ROM/DR+RO/RMAX
E(J,J,NDR)=0.5*VR/DR
J2NDR=NAE+J+NV*(1V2-1+NV*(NDR-1))
E(J,2,NDR)=RO*0.5/DR
F(J,NDR)=RO/DT
GO TO 2000
155 F(J,NDR)=RU
GO TO 2000
200 GO TO (201,202,250,204,205,206),IBCCRJ
201 E(J,J)=1.
202 E(J,J)=0.5
204 E(J,J)=1.
C FROM Z COMP., OHM S LAW, WITH BR=0.=VP, TO O(DR)+O(DT**2)
210 CALL TRANCO(VVARI,TC,DICDV,0)

```



```

G(J,J,1)=BBP
IV8=IV(8)
G(J,IV8,1)=VVR+RES*(1./DRM-1./RR(1))/UO
J81=NAE+J+NV*(IV(8)-1)
E(J,81)=RES/(UO*DRM)
CR=((BBPP-BBP)/DRM+BBP/RR(1))/UO
F(J,1)=CR*RES-EZ
DO 212 L=1,NV
LL=IVV(LL)
G(J,L,1)=G(J,L,1)-CR*DTCDV(4,LL)
212 GO TO 250
F(J,1)=VVR
206 F(J,1)=VVR
GO TO 250
250 GO TO (251,252,2000,254,255),IBCRJ
251 E(J,J,NDR)=0.5
GO TO 2000
252 E(J,J,NDR)=1.
GO TO 2000
C FROM ID CONTINUITY EQUATION WITH RO(NDR+1)=0.=VR(NDR+1)
254 G(J,J,2)=-0.5*ROM/DR+RO/RMAX
IV1=IV(1)
G(J,IV1,2)=1./DT-VRM*0.5/DR+VR/RMAX
E(J,J,NDR)=0.5*RO/DR
J1NDR=NAE+J+NV*NV*(NDR-1)
E(J,1,NDR)=0.5*VR/DR
F(J,NDR)=RO/DT
GO TO 2000
255 F(J,NDR)=VR
GO TO 2000
300 CONTINUE
301 E(J,J,1)=-1.
GO TO 350
350 GO TO (351,352,2000),IBCRJ
351 E(J,J,NDR)=1.
GO TO 2000
352 E(J,J,NDR)=0.5
GO TO 2000
400 CONTINUE
401 E(J,J,1)=1.
GO TO 450
450 GO TO (451,452,2000),IBCRJ
451 E(J,J,NDR)=1.
GO TO 2000
452 E(J,J,NDR)=0.5
GO TO 2000
500 GO TO (501,502),IBCRJ
501 E(J,J,1)=1.

```

5540
5550
5560
5570
5580
5590
5600
5610
5620
5630
5640
5650
5660
5670
5680
5690
5700
5710
5720
5730
5740
5750
5760
5770
5780
5790
5800
5810
5820
5830
5840
5850
5860
5870
5880
5890
5900
5910
5920
5930
5940
5950
5960
5970
5980
5990
6000

502	GO TO 550	6010
	F(J,1)=TI	6020
550	GO TO 550	6030
	IF(IAV(5).EQ.1) GO TO 560	6040
551	GO TO (551,552),IBCRJ	6050
	CONTINUE	6060
	F(J,NDR)=TI	6070
560	GO TO(561,562,563),IBCRJ	6080
561	PRI=I	6090
	ASSUME IBCR(1)=1,2	6100
C	F(J,NDR)=PRI*F(NAF+1+NV*(NDR-1))/RO	
	GO TO 2000	6110
552	CONTINUE	6120
	E(J,NDR)=1.	6130
562	GO TO 2000	6140
	PRI=TI	6150
	E(J,NDR)=1.	6160
	E(J,1,2)=PRI/RO	6170
C	E(J,1,NDR)=...NV*(NDR-1)	6180
	JINDR=NAE+J+NV*(NDR-1)	
	E(J,1,NDR)=-PRI/RO	6190
	GO TO 2000	6200
563	F(J,NDR)=TI	6210
	GO TO 2000	6220
600	GO TO (601,602),IBCRJ	6230
601	E(J,1)=1.	6240
602	GO TO 650	6250
	F(J,1)=TTE	6260
650	GO TO 650	6270
	IF(IAV(6).EQ.1) GO TO 660	6280
651	GO TO (651,652),IBCRJ	6290
	CONTINUE	6300
	F(J,NDR)=TE	6310
	GO TO 2000	6320
660	GO TO (661,662,663),IBCRJ	6330
661	PRE=TE	6340
C	ASSUME IBCR(1)=1,2	6350
	F(J,NDR)=PRE*F(NAF+1+NV*(NDR-1))/RO	
	GO TO 2000	6360
652	CONTINUE	6370
	E(J,NDR)=1.	6380
662	GO TO 2000	6390
	PRE=TE	6400
	E(J,NDR)=1.	6410
	E(J,1,2)=-PRE/RO	6420
C	E(J,1,NDR)=...	6430
	JINDR=NAE+J+NV*(NDR-1)	6440


```

6450 E(J,1,NDR)=-PRE/RO
6460 GO TO 2000
6470 F(J,NDR)=TE
6480 GO TO 2000
6490 CONTINUE
6500 E(J,1,1)=-1.
6510 GO TO 750
6520 GO TO 751,2000,1,IBCRJ
6530 FROM DIV,B=0., TO 0(DR**2)
6540 R=(RR(NDR)+RR(NDR-1))/2.
6550 E(J,J,NDR)=(2.*R-DR)/(2.*R+DR)
6560 IBZ=NDR*(K+(I*(9)-1)*NDZ)
6570 KZ=NDR+K
6580 DZP=RZ(KZ+1)-RZ(KZ)
6590 DZM=RZ(KZ)-RZ(K-1)
6600 F(J,NDR)=- (R*DR/(2.*R+DR))*{(DZM*(VAR(IBZ+NDR)+VAR(IBZ-1+NDR)))/DZP
6610 1-DZP*(VAR(IBZ+NDR)+VAR(IBZ-1+NDR))/DZM)/(DZP*DZM)}
6620 2BZ1+VAR(IBZ-1))/(DZP*DZM)}
6630 GO TO 2000
6640 800 GO TO (801,802,803,850),IBCRJ
6650 801 E(J,1,1)=-1.
6660 GO TO 850
6670 802 IF(BPF*(TIME+DT).GT.0.25) GO TO 810
6680 F(J,1)=BPINIT+(BPMAX-BPINIT)*SIN (2.*PI*BPF*(TIME+DT))
6690 GO TO 850
6700 F(J,1)=BPMAX
6710 810 GO TO 850
6720 FROM AXIAL CURRENT IZ=0. AT BOUNDARY
6730 803 E(J,1,1)=RR(2)/RR(1)
6740 GO TO (851,852,2000,854),IBCRJ
6750 850 GO TO (BPF*(TIME+DT).GT.0.25) GO TO 860
6760 F(J,NDR)=BPINIT+(BPMAX-BPINIT)*SIN (2.*PI*BPF*(TIME+DT))
6770 GO TO 2000
6780 F(J,NDR)=BPMAX
6790 GO TO 2000
6800 FROM AXIAL CURRENT IZ=0. AT BOUNDARY
6810 852 E(J,1,NDR)=RR(NDR-1)/RR(NDR)
6820 GO TO 2000
6830 FROM Z COMP., OHM S LAW, WITH BR=0.=VP, TO 0(DR)+0(DT**2)
6840 854 EZ=0.
6850 CALL TRANCO(VARI,TC,DTCDV,0)
6860 G(J,J,2)=VR+RES*(-1./DR-1./RMAX)/UO
6870 G(J,1V2,2)=BP
6880 E(J,J,NDR)=-RES/(UO*DR)
6890 CR=((BP-BPM)/DR+BP/RMAX)/UO
6900 F(J,NDR)=CR*RES-EZ
6910 DO 867 L=1,NV
6920 LL=IVV(L)
6930
6940
6950
6960
6970
6980
6990
7000
7010
7020
7030
7040
7050
7060
7070
7080
7090
7100
7110
7120
7130
7140
7150
7160
7170
7180
7190
7200
7210
7220
7230
7240
7250
7260
7270
7280
7290
7300
7310
7320
7330
7340
7350
7360
7370
7380
7390
7400
7410
7420
7430
7440
7450
7460
7470
7480
7490
7500
7510
7520
7530
7540
7550
7560
7570
7580
7590
7600
7610
7620
7630
7640
7650
7660
7670
7680
7690
7700
7710
7720
7730
7740
7750
7760
7770
7780
7790
7800
7810
7820
7830
7840
7850
7860
7870
7880
7890
7900
7910
7920
7930
7940
7950
7960
7970
7980
7990
8000
8010
8020
8030
8040
8050
8060
8070
8080
8090
8100
8110
8120
8130
8140
8150
8160
8170
8180
8190
8200
8210
8220
8230
8240
8250
8260
8270
8280
8290
8300
8310
8320
8330
8340
8350
8360
8370
8380
8390
8400
8410
8420
8430
8440
8450
8460
8470
8480
8490
8500
8510
8520
8530
8540
8550
8560
8570
8580
8590
8600
8610
8620
8630
8640
8650
8660
8670
8680
8690
8700
8710
8720
8730
8740
8750
8760
8770
8780
8790
8800
8810
8820
8830
8840
8850
8860
8870
8880
8890
8900
8910
8920
8930
8940
8950
8960
8970
8980
8990
9000

```



```

867 G(J,L,2)=G(J,L,2)-CR*DTCDV(4,LL)
      F(J,NDR)=F(J,NDR)-CR*DTCDV(4,LL)*VARI(LL)
900 GO TO 2000
901 GO TO (901,902,903), IBCRRJ
902 E(J,J,1)=1.
      GO TO 950
C FROM PHI COMP., OHM S LAW, WITH BR=0., TO O(DR)+O(DT**2)
902 EP=0.
910 CALL TRNCO(VVARI,TC,DTCDV,0)
      G(J,J,1)=-VVR-RES/(DR*UO)
      G(J,IV2,1)=-BBZ
      E(J,J,1)=-RES/(DR*UO)
      CR=-(BBZP-BBZ)/(DR*UO)
      F(J,1)=CR*RES-EP
      DO 912 L=1,NV
      LL=IVV(L)
      G(J,L,1)=G(J,L,1)-CR*DTCDV(4,LL)
      F(J,1)=F(J,1)-CR*DTCDV(4,LL)*VVARI(LL)
912 GO TO 950
903 F(J,1)=BBZ
      GO TO 950
950 GO TO (951,952,953,954,955), IBCRJ
951 IF(NDIM.EQ.2) GO TO 961
      IF(BZF*(TIME+DT).GT.0.25) GO TO 960
      F(J,NDR)=BZINIT+(BZMAX-BZINIT)*SIN(2.*PI*BZF*(TIME+DT))
      GO TO 2000
960 F(J,NDR)=BZMAX
      GO TO 2000
961 IF(BZF*(TIME+DT).GT.0.25) GO TO 962
      ZK=RZ(NDR+K)
      F(J,NDR)=BZINIT+(BZMAX*(BRAT+1.+(BRAT-1.)*SIN(PI*(ZK
      /ZMAX-0.5)
      )/(BRAT*2.)-BZINIT)*SIN(2.*PI*BZF*(TIME+DT))
      GO TO 2000
962 ZK=RZ(NDR+K)
      F(J,NDR)=BZMAX*(BRAT+1.+(BRAT-1.)*SIN(PI*(ZK /ZMAX-0.5)))/(BRAT
      1*2.)
      GO TO 2000
C FROM PHI COMP., OHM S LAW, WITH BR=0., TO O(DR)+O(DT**2)
952 EP=EP+MAX*SIN(2.*PI*EPF*(TIME+DT))
963 CALL TRNCO(VARI,TC,DTCDV,0)
      G(J,J,2)=-VR+RES/(DR*UO)
      G(J,IV2,2)=-BZ
      E(J,J,NDR)=RES/(DR*UO)
      CR=-(BZ-BZM)/(DR*UO)
      F(J,NDR)=CR*RES-EP
      DO 965 L=1,NV
      LL=IVV(L)
      G(J,L,2)=G(J,L,2)-CR*DTCDV(4,LL)

```

6890
6900
6910
6920
6930
6940

6950
6960
6970
6980
6990
7000
7010
7020
7030
7040
7050
7060
7070
7080
7090
7100
7110
7120
7130
7140
7150
7160
7170
7180
7190
7200
7210
7220
7230
7240

7250
7260
7270
7280
7290
7300
7310
7320
7330
7340


```

965 F(J,NDR)=F(J,NDR)-CR*DTCDV(4,LL)*VARI(LL)
966 GO TO 2000
967 EP=0.
968 GO TO 963
969 THETA=ARSIN(BZINIT/BZMAX)
970 THETA=2.*PI*BZF*(TIME+DT)+THETA
971 IF(THETA.GT.PI/2.) GO TO 966
972 F(J,NDR)=BZMAX*SIN(THETA)
973 GO TO 2000
974 F(J,NDR)=BZMAX
975 GO TO 2000
976 F(J,NDR)=BZ
977 GO TO 2000
978 CONTINUE
979 GO TO (1001,1002,1050),IBCRJ
980 CONTINUE
981 IAVP1=IAV(10)+1
982 GO TO(1011,1011,1002),IAVP1
983 E(J,1)=1.
984 GO TO 1050
985 E(J,1)=-1.
986 GO TO 1050
987 GO TO (1051,1052,2000),IBCRJ
988 CONTINUE
989 IF(INDIM.EQ.2) GO TO 1061
990 IF(BZFF*(TIME+DT).GT.0.25) GO TO 1060
991 BZ=BZINIT+(BZMAX-BZINIT)*SIN(2.*PI*BZF*(TIME+DT))
992 GO TO 1063
993 BZ=BZMAX
994 GO TO 1063
995 ZK=RZ(NDR+K)
996 IF(BZFF*(TIME+DT).GT.0.25) GO TO 1062
997 BZ=BZINIT+(BZMAX*(BRAT+1.+(BRAT-1.)*SIN(PI*(ZK/ZMAX-0.5))))/(BRAT+1.)*SIN(PI*(ZK/ZMAX-0.5))
998 GO TO 1063
999 BZ=BZMAX*(BRAT+1.+(BRAT-1.)*SIN(PI*(ZK/ZMAX-0.5)))/(BRAT+1.)*SIN(PI*(ZK/ZMAX-0.5))
1000 GO TO 1063
1001 IAVP1=IAV(10)+1
1002 GO TO (1065,1066,1067),IAVP1
1003 E(J,NDR)=1.
1004 GO TO 2000
1005 E(J,NDR)=RR(NDR-1)/RR(NDR)
1006 GO TO 2000
1007 E(J,NDR)=RR(NDR-1)*RR(NDR-1)/RR(NDR)/RR(NDR)
1008 GO TO 2000
1009 IF(IAV(10).EQ.1) GO TO 1064
1010 E(J,NDR)=1.
1011 F(J,NDR)=DR*(RR(NDR)-0.5*DR)*BZ

```

7350
7360
7370
7380
7390
7400
7410
7420
7430
7440
7450
7460
7470
7480
7490
7500
7510
7520
7530
7540
7550
7560
7570
7580
7590
7600
7610
7620
7630
7640
7650
7660
7670
7680
7690
7700
7710
7720
7730
7740
7750
7760
7770
7780
7790
7800
7810
7820


```

1064      GO TO 2000
          E(J,J,NDR)=1./(2.*DR/RR(NDR))+1.)
          F(J,NDR)=BZ/(2.*RR(NDR)/DR)
2000      CONTINUE
          RETURN
          END

SUBROUTINE BCSETS
COMMON/C3/ DA(175), DADT(25), VAR(3500)
EQUIVALENCE (DA(1),NDIM),(DA(9),NDR),(DA(10),NDZ)
NNNN=NDZ+1-NDIM
CALL BOUNDY(0,NNNN,NDIM,1)
RETURN
END

SUBROUTINE BCRSHK
COMMON/C3/ DA(175), DADT(25), VAR(3500)
EQUIVALENCE (DA(1),NDIM),(DA(9),NDR),(DA(10),NDZ)
NNNN=NDZ+1-NDIM
CALL BOUNDY(1,NNNN,NDIM,1)
RETURN
END

SUBROUTINE BCZ(K)
DIMENSION VARIM(10), VVARI(10), VVARIP(10)
COMMON/C3/ DA(175), DADT(25), VAR(3500)
COMMON/C4/ IBCRR(10), IBCR(10), IBCZZ(10), IBCZ(10), ISHK(10), NZP(3),
1 IMPSHK,JSTEP
COMMON/C5/ G(9,9,2)
COMMON/C6/ ISTEP,IV(10),IVV(10),IAV(10)
COMMON/C7/ A(16,10),B(10,10),D(10,10),V(10),COL(10)
COMMON/C9/ VARI(2M(10)),VARIM(10),VARI(10),VARIRP(10),VARIZP(10),
1 DVDR(10),DVDZ(10),D2VDZ(10),D2VDZ2(10)
COMMON/C20/ NDA,NRZ,NIV,NAT,NAU,NAF,NAE
COMMON/C21/ RZ(1150)
DIMENSION E(10),F(10)
EQUIVALENCE (DA(1),NDIM),(DA(9),NDR),(DA(10),NDZ),
1 (DA(16),NV),(DA(25),RMIN),(DA(29),ROMIN),
2 (DA(30),TRO,XRO),(DA(31),BZINIT),(DA(32),BZMAX,EPMAX),
3 (DA(33),BZF,EPF),(DA(34),BRAT),(DA(35),BPINIT),
4 (DA(36),BPMAX),(DA(37),BPF)
EQUIVALENCE (DADT(1),TIME)
EQUIVALENCE (VAR,E,F)
EQUIVALENCE (VARI(1),RO),(VARI(2),VR),(VARI(3),VP),(VARI(4),VZ),

```

7830
7840
7850
7860
7870
7880
7890

7900
7910
7920
7930
7940
7950
7960

7970
7980
7990
8000
8010
8020
8030

8040
8050
8060
8070
8080
8090
8100
8110
8120
8130
8140
8150
8160
8170
8180
8190
8200
8210
8220
8230
8240


```

1      (VARI(5),TI),(VARI(6),TE),(VARI(8),BR),(VARI(8),BP),
2      (VARI(9),BZ),(VARI(10),SI)
EQUIVALENCE
1      (VARI(1),RRQ,VARI(1)),(VARI(2),VVR),(VARI(3),VVP)
2      (VARI(4),VVZ),(VARI(5),TI),(VARI(6),TTE),
3      (VARI(7),BBR),(VARI(8),BBP),(VARI(9),BBZ),
      (VARI(10),SSI)
EQUIVALENCE
1      (VARI(1),ROM,VARI(2),VRM),(VARI(3),VPM)
2      (VARI(4),VZM),(VARI(5),TIM),(VARI(6),TEM),
3      (VARI(7),BRM),(VARI(8),BPM),(VARI(9),BZM),
      (VARI(10),SIM)
EQUIVALENCE
1      (VARI(1),RRQP,VARI(2),VVRP),(VARI(3),TTIP),
2      (VARI(4),VVPP),(VARI(5),VVZP),(VARI(6),BBRP),(VARI(7),BBPP),
3      (VARI(8),BBZP),(VARI(9),SSIP)
      (VARI(10),SSIP)
DO 10 J=1,NV
JJ=I(VV(J))
VARI(JJ)=VAR(K+NDR*(-1+J*NDZ))
VARI(JJ)=VAR(K+NDR*(-2+J*NDZ))
VARI(JJ)=VAR(K+NDR*(J-1)*NDZ)
VARI(JJ)=VAR(K+NDR*(1+(J-1)*NDZ))
F(NAF+J)=0.(NDZ-1)
JZ=NAF+J+NV*(NDZ-1)
F(JZ)=0.
DO 10 L=1,NV
JL=NAE+J+NV*(L-1)
JLZ=NAE+J+NV*(L-1+NV*(NDZ-1))
E(JL)=0.
E(JLZ)=0.
DO 10 M=1,2
G(J,L,M)=0.
IF(J.EQ.L) G(J,L,M)=1.
CONTINUE
DN=RZ(NDR+NDZ)-RZ(NDR+NDZ-1)
NVP=NV+1
DO 20 J=NVP,10
JJ=I(VV(J))
VARI(JJ)=0.
VARI(JJ)=0.
VARI(JJ)=0.
VARI(JJ)=0.
DO 2000 J=1,NV
JJ=NAE+J+NV*(J-1)
JJ=NAE+J
JJNDZ=NAE+J+NV*(J-1+NV*(NDZ-1))
JNDZ=NAE+J+NV*(NDZ-1)
JJ=I(VV(J))
IBCZJ=IBCZ(JJ)
IBCZZJ=IBCZZ(JJ)

```



```

100 GO TO (100,200,300,400,500,600,700,800,900,1000), JJ
101 CONTINUE
102 E(J,J,1)=1.
150 GO TO 150
151 GO TO (151,152,153), IBCZJ
152 E(J,J,NDZ)=1.
153 GO TO 2000
154 F(J,NDZ)=ROMIN+(RO-ROMIN)*EXP (-DT/TRO)
155 GO TO 2000
156 F(J,NDZ)=ROMIN+(RO-ROMIN)*EXP {VZ*DT/(XRO*DZ)}
157 GO TO 2000
200 CONTINUE
201 E(J,J,1)=1.
250 GO TO 250
251 GO TO (251,252,253,2000), IBCZJ
252 E(J,J,NDZ)=1.
253 GO TO 2000
254 F(J,J,NDZ)=0.5
255 GO TO 2000
256 E(J,J,NDZ)=-1.
257 GO TO 2000
300 CONTINUE
301 E(J,J,1)=1.
350 GO TO 350
351 GO TO (351,352,2000), IBCZJ
352 E(J,J,NDZ)=1.
353 GO TO 2000
354 E(J,J,NDZ)=-1.
355 GO TO 2000
400 GO TO 2000
401 GO TO (401,450), IBCZJ
402 E(J,J,1)=-1.
403 GO TO 450
450 GO TO (451,452,453), IBCZJ
451 E(J,J,NDZ)=-1.
452 GO TO 2000
453 E(J,J,NDZ)=1.
454 GO TO 2000
455 F(J,J,NDZ)=0.5
456 GO TO 2000
500 CONTINUE
501 E(J,J,1)=1.
550 GO TO 550
551 GO TO (551,552), IBCZJ
552 E(J,J,NDZ)=1.
553 GO TO 2000
554 F(J,NDZ)=1
555 GO TO 2000
600 CONTINUE

```

8730
8740
8750
8760
8770
8780
8790
8800
8810
8820
8830
8840
8850
8860
8870
8880
8890
8900
8910
8920
8930
8940
8950
8960
8970
8980
8990
9000
9010
9020
9030
9040
9050
9060
9070
9080
9090
9100
9110
9120
9130
9140
9150
9160
9170
9180
9190
9200


```

601 E(J J 1)=1.
650 GO TO 650
650 GO TO (661,662),IBCZJ
661 E(J J NDZ)=1.
662 GO TO 2000
662 F(J NDZ)=TE
700 GO TO 2000
700 CONTINUE
701 E(J J 1)=-1.
701 GO TO 750
750 CONTINUE
751 E(J J NDZ)=-1.
800 GO TO 2000
800 CONTINUE
801 E(J J 1)=1.
801 GO TO 850
850 CONTINUE
851 E(J J NDZ)=1.
900 GO TO 2000
900 CONTINUE
901 E(J J 1)=1.
901 GO TO 950
950 CONTINUE
951 E(J J NDZ)=1.
951 GO TO 2000
1000 CONTINUE
1001 E(J J 1)=1.
1001 GO TO 1050
1050 CONTINUE
1051 E(J J NDZ)=1.
1051 GO TO 2000
2000 CONTINUE
      RETURN
      END

```

```

SUBROUTINE BCZOTS
COMMON/C3/ DA(175),DADI(25),VAR(3500)
EQUIVALENCE (DA(1),NDIM),(DA(9),NDR),(DA(10),NDZ)
NNNN=NDR-1
CALL BOUNDY(0,NNNN,2,2)
      RETURN
      END

```

9210
9220
9230
9240
9250
9260
9270
9280
9290
9300
9310
9320
9330
9340
9350
9360
9370
9380
9390
9400
9410
9420
9430
9440
9450
9460
9470
9480
9490
9500
9510
9520
9530
9540

9550
9560
9570
9580
9590
9600
9610

9620
9630
9640
9650
9660
9670
9680

```
SUBROUTINE BCZSHK
COMMON/C3/ DA(175), DADT(25), VAR(3500)
EQUIVALENCE (DA(1),NDIM), (DA(9),NDR), (DA(10),NDZ)
NNNN=NDR-1
CALL BOUNDY(1,NNNN,2,2)
RETURN
END
```

9690

```
SUBROUTINE BEXINT(IBEDAT)
THIS SUBROUTINE USES FILE DATA
THIS SUBROUTINE USES INPUT CARDS NUMBER 10.
REAL*8 IDABEV,COMT1,COMT2,COMT3,IDA
COMMON/C3/ DA(175), DADT(25), VAR(3500)
COMMON/C14/ BZEINT,BZEMAX,BZEF
COMMON/C30/ INP,IOUT1,IOUT2,INP2,IOUT4
DIMENSION IDA(40),ADDA(40)
EQUIVALENCE(DA(20),NADDA), (DA(71),IDAL1), (DA(26),ADDA(1))
DATA IDABEV/.8E VERS=.,COMT1/. BZEINT./,COMT2/. BZEMAX./,
1 COMT3/. BZEF./
```

10 IF(IDF DAT.EQ.1) GO TO 15
READ(INP,10) BZEINT,BZEMAX,BZEF
FORMAT(5X,7E10.3)
IF(BZEINT.EQ.BZEMAX) BZEF=0.

```
IDA(NADDA+1)=IDABEV
ADDA(NADDA+1)=1.
IDA(NADDA+2)=COMT1
ADDA(NADDA+2)=BZEINT
IDA(NADDA+3)=COMT2
ADDA(NADDA+3)=BZEMAX
IDA(NADDA+4)=COMT3
ADDA(NADDA+4)=BZEF
NADDA=NADDA+4
GO TO 25
```

15 CONTINUE
DO 20 J=1,NADDA

```
IF(IDA(J).NE.IDABEV) GO TO 20
BZEINT=ADDA(J+1)
BZEMAX=ADDA(J+2)
BZEF=ADDA(J+3)
GO TO 25
```

20 CONTINUE
25 RETURN
END

C C


```

SUBROUTINE BEXT(J,K)
COMMON/C1/DUMDUM(17),PI
COMMON/C3/DA(175),DADT(25),VAR(3500)
COMMON/C6/ISTEP,IV(10),IIV(10),IIV(10),IIV(10)
COMMON/C11/BZSTEP(4),BERM(4),BE(4),BERP(4),BEPZM(4),BEPZ(4),BEPDR(4),BEPZP(4),DBEDR(4),DBEDZ(4),
1 DBEPRM(4),RER(4),BEPDZ(4)
2 DBEPRD(4),DBEPRDZ(4)
COMMON/C14/BZEINT,BZEMAX,BZEF
COMMON/C21/RZ(150)
COMMON/C26/BERMZP(04),BERPZP(04),BERMZM(04),BERPZM(04)
DIMENSION CEB(16),EB(56),RR(10)
EQUIVALENCE (CEB,BERMZP)
EQUIVALENCE (DADT(1),DT),(DA(16),NV),(DA(24),IBE)
EQUIVALENCE (EB(1),BEZM(1))
EQUIVALENCE (RZ,RR)
DO 5 L=1,56
5 EB(L)=0.
DO 6 L=1,16
6 CEB(L)=0.
IF(IBE.NE.1) RETURN
BZ=BZEINT
BZP=BZEINT
IF(BZEINT.EQ.0.) GO TO 10
BZ=BZEINT+(BZEMAX-BZEINT)*SIN(2.*PI*BZEF*TIME)
BZP=BZEINT+(BZEMAX-BZEINT)*SIN(2.*PI*BZEF*(TIME+DT))
10 CONTINUE
DO 15 L=1,5
LEB=3+4*(L-1)
LEBP=23+4*(L-1)
EB(LEB)=BZ
EB(LEBP)=BZP
15 CONTINUE
DO 17 L=1,4
LEB=3+4*(L-1)
LEBP=23+4*(L-1)
CEB(LEB)=BZ
17 IF(IV(10).GT.NV) GO TO 25
IAPV1=IAPV(10)+1
GO TO(20,21,22),IAPV1
20 Q=0.5*RR(J)*RR(J)
Q=RR(J)
QP=0.5*RR(J+1)*RR(J+1)
QM=0.5*RR(J-1)*RR(J-1)
GO TO 23
21 Q=0.5
Q=Q.0
QP=Q
QM=Q

```


10500
10510
10520
10530
10540
10550
10560
10570
10580
10590
10600
10610
10620
10630
10640
10650
10660
10670
10680
10690
10700
10710
10720

```

GO TO 23
Q=0.5*RR(J)
DQ=0.5
QM=0.5*RR(J+1)
BEZM(4)=Q*BZ
BEZM(4)=BEZM(4)
BEZP(4)=BEZM(4)
DBEDR(4)=DQ*BZ
BERP(4)=QP*BZ
BERMZP(4)=QM*BZ
BERMZM(4)=BERM(4)
BERMZM(4)=BERM(4)
BERPZP(4)=BERP(4)
BERPZM(4)=BERP(4)
BEPZM(4)=Q*BZP
BEP(4)=BEPZM(4)
BEPZP(4)=BEPZM(4)
DBEPDR(4)=DQ*BZP
BERPRP(4)=QP*BZP
BERPRM(4)=QM*BZP
RETURN
END
25

```

10730
10740
10750
10760
10770
10780
10790
10800
10810
10820
10830
10840
10850
10860
10870
10880
10890
10900
10910
10920
10930
10940
10950

```

SUBROUTINE BFRMSI(J,K,BR,BZ)
COMMON/C3/ DA(175),DADT(25),VAR(3500)
COMMON/C6/ STEP,IV(10),IAV(10)
COMMON/C21/ RZ(150)
EQUIVALENCE (DA(1),NDIM),(DA(9),NDR),(DA(10),NDZ)
EQUIVALENCE (DSIDR,DAPDR,DSBDR),(SI,AP,SB),(DSIDZ,DAPDZ,OSBDZ)
JSI=J+NDR*(K-1+NDZ*(IV(10)-1))
SI=VAR(JSI)
IAVP1=IAV(10)+1
R=RZ(J)
DRP=RZ(J+1)-R
DRM=R-RZ(J-1)
IF(J.LE.2) GO TO 10
IF(J.EQ.NDR) GO TO 20
DSIDR=(DRM*(VAR(JSI+1)-SI)/DRP+DRP*(SI-VAR(JSI-1))/DRM)/(DRP+DRM)
GO TO (1,2,3),IAVP1
1 BZ=DSIDR/R
GO TO 30
2 BZ=R*OSBDR+2.*SB
GO TO 30
3 BZ=DAPDR+AP/R
GO TO 30
10 CONTINUE

```


C ASSUMED BOUNDARY IS AT ORIGIN, BZ IS UNIFORM

```

GO TO (11,12,13), IAVPI
11 BZ=2.*SI/R/R
GO TO 30
12 BZ=2.*S8
GO TO 30
13 BZ=2.*AP/R
GO TO 30
20 DSIDR=(SI-VAR(JSI-1))/DRM
GO TO (1,2,3), IAVPI
30 BR=0.
IF (NDIM.EQ.1) RETURN
Z=RZ(NDR+K)
DZP=RZ(NDR+K+1)-Z
DZM=Z-RZ(NDR+K-1)
IF (K.EQ.1) GO TO 40
IF (K.EQ.NDZ) GO TO 50
DSIDZ=(DZM*(VAR(JSI+NDR)-SI))/DZP+DZP*(SI-VAR(JSI-NDR))/DZM)/(DZP+D
1ZM)
35 GO TO (36,37,38), IAVPI
36 BR=-DSIDZ/R
RETURN
37 BR=-R.*DSBZ
RETURN
38 BR=DAPDZ
RETURN
40 DSIDZ=(VAR(JSI+NDR)-SI)/DZP
GO TO 35
50 DSIDZ=(SI-VAR(JSI-NDR))/DZM
GO TO 35
END

```

SUBROUTINE BOUNDY(IENT,NNNN,NNN,ID)

```

INTEGER COL
COMMON/C3/ DA(175), DADT(25), VAR(3500)
COMMON/C5/ G(9,9,2)
COMMON/C7/ A(10,10), B(10,10), D(10,10), V(10), COL(10)
COMMON/C20/ NDA, NRZ, NTV, NAT, NAU, NAF, NAE
DIMENSION E(10), F(10)
EQUIVALENCE (DA(1),NDIM), (DA(9),NDR), (DA(10),NDZ), (DA(16),NV)
EQUIVALENCE (VAR,E,F)
DO 110 NN=NNN,NNNN
IF (ID.EQ.1) CALL BCR(NN)
IF (ID.EQ.2) CALL BCZ(NN)
DO 110 L=1,2
GO TO 8
IF (ID.EQ.2) GO TO 8
J=NDR*(L-1)+2-L

```

6


```

1410
1420
1430
1440
1450
1460
1470
1480
1490
1500
1510
1520
1530
1540
1550
1560
1570
1580
1590
1600
1610
1620
1630
1640
1650
1660
1670
1680
1690
1700
1710
1720
1730
1740
1750
1760
1770
1780
1790
1800
1810
1820
1830
1840
1850
1860
1870
1880

IB=J
K=NN
JIN=NDR*(L-1)+5-3*L
KIN=K
GO TO 10
8 K=NDZ*(L-1)+2-L
IB=K
J=NN
KIN=NDZ*(L-1)+5-3*L
JIN=J
10 IF(IENT.EQ.0) GO TO 40
DO 30 M=1,NV
DO 20 N=1,NV
NE=NAE+M+NV*(N-1+NV*(IB-1))
IF((M.NE.N).AND.(G(M,N,L).NE.0.)).OR.(E(NE).NE.0.)) GO TO 30
20 CONTINUE
MIB=NAE+M+NV*(IB-1)
JKM=J+NDR*(K-1+(M-1)*NDZ)
F(MIB)=VAR(JKM)
30 CONTINUE
40 CONTINUE
DO 50 M=1,NV
V(M)=F(NAF+M+NV*(IB-1))
DO 50 N=1,NV
V(M)=V(M)+E(NAE+M+NV*(N-1+NV*(IB-1)))*VAR(JIN+NDR*(KIN-1+(N-1)*NDZ
1))
50 A(M,N)=G(M,N,L)
DO 60 M=1,NV
DO 60 N=1,NV
IF((N.NE.M).AND.(A(M,N).NE.0.)) GO TO 80
60 CONTINUE
DO 70 M=1,NV
JKM=J+NDR*(K-1+(M-1)*NDZ)
70 VAR(JKM)=V(M)
GO TO 110
80 CALL TRIANG(1)
DO 90 M=1,NV
JKM=J+NDR*(K-1+(COL(M)-1)*NDZ)
90 VAR(JKM)=V(M)
110 CONTINUE
IF(NDIM.EQ.1) GO TO 130
DO 120 L=1,NV
JK=1+NDR*NDZ*(L-1)
VAR(JK)=VAR(JK+1)+VAR(JK+NDR)-VAR(JK+1+NDR)
JK=1+NDR*(-1+L*NDZ)
VAR(JK)=VAR(JK+1)+VAR(JK+NDR)-VAR(JK+1+NDR)
JK=NDR*(1+NDZ*(L-1))
VAR(JK)=VAR(JK-1)+VAR(JK+NDR)-VAR(JK-1+NDR)

```



```

10 JK=L*NDR*NDZ
110 VAR(JK)=VAR(JK-1)+VAR(JK-NDR)-VAR(JK-1-NDR)
120 CONTINUE
130 RETURN
END

```

11890
11900
11910
11920
11930

```

SUBROUTINE BUFFER(IRORW,IARRAY,ITP)
COMMON/C3/ DA(175),DADT(25),VAR(3500)
COMMON/C20/ IDUM(2),NTV
COMMON/C21/ RZ(150)

```

11940
11950
11960
11970
11980
11990
12000
12010
12020
12030
12040
12050
12060
12070
12080
12090
12100
12110
12120
12130
12140
12150
12160
12170
12180
12190
12200
12210
12220
12230
12240
12250
12260
12270
12280
12290
12300
12310

```

GO TO (10,20,30),IARRAY
DO 12 J=1,7
10 K1=25*(J-1)+1

```

```

K2=K1+24
IF(IRORW.EQ.1) GO TO 11
READ(ITP)(DA(K),K=K1,K2)
GO TO 12

```

```

11 WRITE(ITP)(DA(K),K=K1,K2)
12 CONTINUE

```

```

DO 22 J=1,6
20 K1=25*(J-1)+1

```

```

K2=K1+24
IF(IRORW.EQ.1) GO TO 21
READ(ITP)(RZ(K),K=K1,K2)
GO TO 22

```

```

21 WRITE(ITP)(RZ(K),K=K1,K2)
22 CONTINUE

```

```

DO 32 J=1,NNTV/25
30 NNTV=NTV/25
NNTV2=NTV-25*NNTV

```

```

GO 32 J=1,NNTV
K1=25*(J-1)+1
K2=K1+24
IF(IRORW.EQ.1) GO TO 31

```

```

READ(ITP)(VAR(K),K=K1,K2)
GO TO 32

```

```

31 WRITE(ITP)(VAR(K),K=K1,K2)
32 CONTINUE

```

```

K1=25*NNTV+1
IF(IRORW.EQ.1) GO TO 33
READ(ITP)(VAR(K),K=K1,NTV)
GO TO 34

```

```

33 WRITE(ITP)(VAR(K),K=K1,NTV)
34 RETURN
END

```



```

SUBROUTINE CHECK(IRR,IZZ,MODE)
THIS SUBROUTINE USES FILES SCRAT AND MHDOUT
INTEGER I1
COMMON/C3/ JCHK(5)
COMMON/C3/ DA(175), DADT(25), VAR(3500)
COMMON/C6/ ISTEP, IV(10), IJV(10), IAV(10)
COMMON/C1/ A(10,10), B(10,10), C(10,10), V(10), COL(10)
COMMON/C20/ NDA, NRZ, NTV, NAT, NAU, NAF, NAE
COMMON/C30/ INP, IOUT1, IOUT2, INP2, IOUT4
DIMENSION BC(10,10,2)
DIMENSION U(10), E(10), F(10)
DIMENSION VV(10)
DIMENSION BC(1), B(11)
EQUIVALENCE (COL(1),IR), (COL(2),IZ), (COL(3),INDEX)
EQUIVALENCE (DA(1),NDIM), (DA(9),NDR), (DA(10),NDZ), (DA(11),DT),
EQUIVALENCE (DA(16),NV), (DA(52),MATALT)
1 EQUIVALENCE (DADT(1),TIME), (DADT(2),NNDT), (DADT(5),MMAT)
EQUIVALENCE (VAR,U,E,F)
I1=IOUT4
I1=IRR
I2=IZZ
GO TO (1001,1002,1003),MODE
CONTINUE
GO TO (11,12,13,14),ISTEP
11 I1=2
I2=NDIM-1
INDEX=IR
GO TO 15
12 I1=1
I2=2
INDEX=IZ
GO TO 15
13 I1=NDR-1
I2=NDIM-1
INDEX=IR
GO TO 15
14 I1=1
I2=NDZ-1
INDEX=IZ
GO TO 25
15 IF((JCHK(1),EQ,0) GO TO 25
IF((IR,EQ,I1).OR.(IZ,EQ,I2)) WRITE(IOUT2 ,76)
CALL SPECHK(IRR,IZZ)
WRITE(IOUT2 ,20) IR,IZ,TIME,NNDT,MMAT,MATALT
FORMAT(/,5X,'A,B,C,V AT ',I3,' ',I3,' '),TIME=' ,E13.6',' , TIME STEP
1= ,I4,5X,' MATRICES VERSION',I3,' ',I1)
DO 22 J=1,NV
22 WRITE(IOUT2 ,95) (A(J,K),K=1,NV)
WRITE(IOUT2 ,97)

```



```

DO 23 J=1,NV
WRITE(IOUT2,95) (B(J,K),K=1,NV)
DO 24 J=1,NV
WRITE(IOUT2,95) (C(J,K),K=1,NV)
WRITE(IOUT2,95) (V(J),J=1,NV)
NEXT=1
IF(INDEX.EQ.1Z) NXT=NDR
WRITE(IOUT2,501)
FORMAT(/,5X,'A.U.B.U(+),C.U(-) IF EXPLICIT')
DO 505 J=1,NV
DO 503 K=1,NV
VK=IR+NDR*(IZ-1+NDZ*(K-1))
VV(K)=A(J,K)*VAR(JK)
WRITE(IOUT2,95) (VV(K),K=1,NV)
505 CONTINUE
DO 510 J=1,NV
WRITE(IOUT2,97)
DO 505 K=1,NV
JK=IR+NDR*(IZ-1+NDZ*(K-1))
VV(K)=B(J,K)*VAR(JK+NXT)
WRITE(IOUT2,95) (VV(K),K=1,NV)
510 CONTINUE
DO 515 J=1,NV
WRITE(IOUT2,97)
DO 513 K=1,NV
JK=IR+NDR*(IZ-1+NDZ*(K-1))
VV(K)=C(J,K)*VAR(JK-NXT)
WRITE(IOUT2,95) (VV(K),K=1,NV)
515 CONTINUE
DO 26 J=1,NV
VV(J)=V(J)
DO 27 K=1,NV
JK=IR+NDR*(IZ-1+NDZ*(K-1))
VV(J)=VV(J)-A(J,K)*VAR(JK)-B(J,K)*VAR(JK+NXT)+C(J,K)*VAR(JK-NXT)
IF(J.NE.K) GO TO 27
VV(J)=VV(J)+VAR(JK)/DT
27 CONTINUE
VV(J)=VV(J)*DT
26 IF(MMAT.NE.2) GO TO 32
J=1
DO 31 JJ=2,6
IF(IV(JJ).GT.NV) GO TO 31
J=J+1
JK1=IR+NDR*(IZ-1)
JK=IR+NDR*(IZ-1+NDZ*(J-1))
IF((JJ.EQ.5).OR.(JJ.EQ.6)).AND.((IAV(JJ).EQ.1)) GO TO 31

```



```

VV(J)=VV(J)-VAR(JK)
VV(J)=VV(J)+VAR(JK)*VV(1)
VV(J)=VV(J)/VAR(JK1)
31 CONTINUE
32 CONTINUE
28 WRITE(IOUT2,28)
FORMAT(/,5X,'NEW VARIABLES IF EXPLICIT')
WRITE(IOUT2,95) (VV(J),J=1,NV)
25 CONTINUE
25 IF((JCHK(4).EQ.0).AND.(JCHK(5).EQ.0)) GO TO 37
35 WRITE(1) A,B,C,J,COL
37 CONTINUE
RETURN
1002 CONTINUE
GO TO (38,39,38,39), ISTEP
38 NP=NDR
GO TO 41
39 NP=NDZ
41 IF((JCHK(4).EQ.0).AND.(JCHK(5).EQ.0)) GO TO 400
45 NP=NP-1
REWIND 11
DO 120 J=2,NPM
READ(1) A,B,C,J,COL
55 K=INDEX
IF(JCHK(4).EQ.0) GO TO 120
GO TO(51,51,52,52), ISTEP
51 JNN=2
LAST=K-1
INN=1
GO TO 53
52 JNN=1
LAST=K+1
INN=2
53 CONTINUE
DO 57 L=1,NV
DO 57 M=1,NV
57 B(L,M)=-B(L,M)
DO 65 L=1,NV
DO 65 M=1,NV
DO 60 N=1,NV
DO 60 M=1,NV
A(L,M)=A(L,M)-BC(L,N,JNN)*E(NAE+N*NV*(M-1+NV*(LAST-1)))
60 A(L,M)=V(L)+BC(L,M,JNN)*F(NAE+M*NV*(LAST-1))
65 IF(J.EQ.2) WRITE(IOUT2,76)
76 FORMAT(1)
WRITE(IOUT2,80) IR,IZ
80 FORMAT(/,2I5)
DO 90 L=1,NV
DO 90 L=1,NV
90 WRITE(IOUT2,95) (BC(L,M,INN),M=1,NV)

```

13270
13280
13290
13300
13310
13320
13330
13340
13350
13360
13370
13380
13390
13400
13410
13420
13430
13440
13450
13460
13470
13480
13490
13500
13510
13520
13530
13540
13550
13560
13570
13580
13590
13600
13610
13620
13630
13640
13650
13660
13670
13680
13690
13700
13710
13720
13730
13740


```

95 FORMAT((7E17.9))
WRITE(IOUT2,97)
DO 91 L=1,NV
91 WRITE(IOUT2,96) (BC(L,M,INN),M=1,NV)
96 FORMAT((7(1X,O16)))
96 FORMAT(7(9X,Z8))
WRITE(IOUT2,97)
97 FORMAT(,1)
WRITE(IOUT2,95) (V(L),L=1,NV)
WRITE(IOUT2,96) (V(L),L=1,NV)
DO 105 L=1,NV
V(L)=0.
DO 105 M=1,NV
B(L,M)=0.
DO 100 N=1,NV
B(L,M)=B(L,M)+A(L,N)*E(NAE+N+NV*(M-1+NV*(K-1)))
105 V(L)=V(L)+A(L,M)*F(NAF+M+NV*(K-1))
WRITE(IOUT2,80) IR,IZ
DO 110 L=1,NV
110 WRITE(IOUT2,95) (B(L,M),M=1,NV)
WRITE(IOUT2,97)
DO 111 L=1,NV
111 WRITE(IOUT2,96) (B(L,M),M=1,NV)
WRITE(IOUT2,97)
WRITE(IOUT2,95) (V(L),L=1,NV)
WRITE(IOUT2,96) (V(L),L=1,NV)
120 CONTINUE
WRITE(IOUT2,175) TIME
400 IF(JCHK(2).EQ.0) GO TO 420
DO 415 L=1,NP
415 L=L+1
GO TO 401,402,401,402,ISTEP
WRITE(IOUT2,405) L,IRR,L,IRR,TIME,NNDT
401 GO TO 403
402 WRITE(IOUT2,405) IRR,L,IRR,L,TIME,NNDT
403 CONTINUE
405 FORMAT(7,5X,'E(',I3,',',I3,')',F(',I3,',',I3,')', TIME=' ,E13.6',,11)
1ME STEP=,14)
DO 419 M=1,NV
DO 407 N=1,NV
V(N)=E(NAE+M+NV*(N-1+NV*(L-1)))
407 V(N)=E(NAE+M+NV*(N-1+NV*(L-1)))
410 WRITE(IOUT2,95) (V(N),N=1,NV)
WRITE(IOUT2,97)
DO 412 M=1,NV
412 V(M)=F(NAF+M+NV*(L-1))
WRITE(IOUT2,95) (V(M),M=1,NV)
415 CONTINUE
420 CONTINUE

```

13750
13760
13770
13780

13790
13800
13810
13820
13830
13840
13850
13860
13870
13880
13890
13900
13910
13920
13930
13940
13950
13960
13970
13980
13990
14000
14010
14020
14030
14040
14050
14060
14070
14080
14090
14100
14110
14120
14130
14140
14150
14160
14170
14180
14190
14200
14210


```

1003 RETURN
      CONTINUE
      IF(JCHK(5).EQ.0) GO TO 176
130  CONTINUE
      REWIND T1
      KK=2
      KKK=JCHK(1)
      IF(JCHK(5).NE.0) KKK=NPM
      DO 170 K=2,NPM
        READ(T1) A,B,C,J,COL
140  J=INDEX
      IF(K.EQ.2) WRITE(IOUT2,76)
      WRITE(IOUT2,80) IRR,IZ
      WRITE(IOUT2,95) (V(L),L=1,NV)
      WRITE(IOUT2,96) (V(L),L=1,NV)
      DO 160 L=1,NV
        V(L)=0.
      DO 160 N=1,NV
        MJ=NAU+M+NV*(J-1)
160  V(L)=V(L)+A(L,M)*U(MJ)+B(L,M)*U(MJ+NV)-C(L,M)*U(MJ-NV)
      WRITE(IOUT2,95) (V(L),L=1,NV)
      WRITE(IOUT2,96) (V(L),L=1,NV)
170  CONTINUE
      WRITE(IOUT2,175) TIME
175  FORMAT(/,' CALCULATIONS CHECKED AT TIME',E15.6)
176  REWIND T1
      CONTINUE
      IF(JCHK(3).EQ.0) GO TO 200
      GO TO(181,182,181,182),ISTEP
181  NP=NDR 183
182  NP=NDZ
183  CONTINUE
      DO 185 L=1,NP
      GO TO(190,191,190,191),ISTEP
190  WRITE(IOUT2,193) L,IRR,TIME,NNDT
      GO TO 192
191  WRITE(IOUT2,193) IRR,L,TIME,NNDT
192  CONTINUE
193  FORMAT(/,5X,'U(',I3,',',I3,',', TIME=' ,E13.6,', TIME STEP=' ,I4)
      MJ=NAU+NV*(L-1)+1
      MM=NAU+NV*L
      WRITE(IOUT2,95) (U(M),M=MM,MMM)
      WRITE(IOUT2,96) (U(N),M=MM,MMM)
185  CONTINUE
186  CONTINUE
      RETURN
      END

```



```

SUBROUTINE INI1
COMMON/C3/ DA(175),DADT(25),VAR(3500)
COMMON/C4/ IBCRR(10),IBCR(10),IBZZ(10),IBZ(10),ISHK(10),NZP(3),
      IMPSHK,JSTEP
1 COMMON/C6/ISTEP,IV(10),IVV(10),IAV(10)
COMMON/C21/RZ(150)
DIMENSION RR(10)
EQUIVALENCE (DA(9),NDR),(DA(10),NDZ),(DA(12),DR),(DA(13),DZ),
      (DA(16),NV),(DA(24),IBE),(DA(25),RMIN),(DA(27),ROINIT),
      (DA(28),TINIT),(DA(29),ROMINI),(DA(31),BZINIT),
      (DA(35),BPINIT),(DA(67),INITCON)
2
3
EQUIVALENCE (RZ,RR)
INITCON=1
DO 50 L=1,NDR
DO 50 M=1,NDZ
JRO=J+NDR*(K-1)+NDZ*(IV(2)-1)
JVR=J+NDR*(K-1)+NDZ*(IV(3)-1)
VAR(JRO)=ROINIT
VAR(JVR)=0
IF(IV(3).GT.NV) GO TO 46
JVP=J+NDR*(K-1)+NDZ*(IV(3)-1)
VAR(JVP)=0
IF(IV(4).GT.NV) GO TO 47
JVZ=J+NDR*(K-1)+NDZ*(IV(4)-1)
VAR(JVZ)=0
46 CONTINUE
JTE=J+NDR*(K-1)+NDZ*(IV(5)-1)
JTE=J+NDR*(K-1)+NDZ*(IV(6)-1)
VAR(JTE)=TINIT
VAR(JTE)=TINIT
IF(IV(7).GT.NV) GO TO 48
JBR=J+NDR*(K-1)+NDZ*(IV(7)-1)
VAR(JBR)=0
48 JBP=J+NDR*(K-1)+NDZ*(IV(8)-1)
VAR(JBP)=0
IF(IBCRR(8).EQ.2) VAR(JBP)=BPINIT*RR(1)/RR(L)
49 CONTINUE
IF(IV(9).GT.NV) GO TO 35
JBZ=J+NDR*(K-1)+NDZ*(IV(9)-1)
VAR(JBZ)=BZINIT
35 IF(IV(10).GT.NV) GO TO 50
JSI=L+NDR*(M-1)+(IV(10)-1)*NDZ)
IAVP1=IAV(10)+1
GO TO(51,52,53),IAVP1
51 Q=0.5*RR(L)*RR(L)

```

14700

14710

14720

14730

14740

14750

14760

14770

14780

14790

14800

14810

14820

14830

14840

14850

14860

14870

14880

14890

14900

14910

14920

14930

14940

14950

14960

14970

14980

14990

15000

15010

15020

15030

15040

15050

15060

15070

15080

15090

15100

15110

15120

15130

15140

15150


```

52 GO TO 54
53 Q=0.5
54 GO TO 54
55 Q=0.5*RR(L)
56 GO TO 54
57 VAR(USI)=Q*BZINIT
58 CONTINUE
59 RETURN
60 END

```

```

C
C
SUBROUTINE INIT4
THIS SUBROUTINE USES FILE DATA
THIS SUBROUTINE USES INPUT CARD NUMBER 12A.
IMPLICIT REAL*8 (C)
REAL COSPHI
REAL*8 IDA
COMMON/C3/ IBCRR(10), IBCZ(10), IBCZZ(10), IBCZ(10), ISHK(10), NZP(3),
COMMON/C4/ IMPSHK, JSTEP
1 COMMON/C6/ ISTEP, IV(10), IVV(10), IAV(10)
COMMON/C21/RZ(150)
COMMON/C30/ INP, IOUT1, IOUT2, INP2, IOUT4
DIMENSION IDA(40), ADDA(40)
EQUIVALENCE
  (DA(16), NV), (DA(20), NDR), (DA(10), NDZ), (DA(12), DR), (DA(13), DZ),
  (DA(26), ADDA(1)), (DA(27), ROINIT), (DA(28), TINIT),
  (DA(29), ROMINI), (DA(31), BZINIT), (DA(35), BPINIT),
  (DA(34), IPLAVC), (DA(65), ZMAX), (DA(66), RMAX),
  (DA(67), INTCON), (DA(71), IDA(1))
EQUIVALENCE (RZ, RR)
DATA COM1 1/, VO=1/, COM2 2/, XO=1/, COM3 3/, EXP=1/,
COM4 4/, VEQO=1/, COM5 5/, VEQVO=1/, COM6 6/, VEQMAX=1/,
COM7 7/, TMIN=1/, COM8 8/, INIT OP=1/
INTCON=4
READ(INP, 5) IOP, VO, XO, TMIN, EXPL, VEQO, VEQVO, VEQMAX
5 FORMAT(4X, 11, 7F10.3)
ADDA(NADDA+1)=TMIN
ADDA(NADDA+2)=VO
ADDA(NADDA+3)=XO
ADDA(NADDA+4)=EXPL
ADDA(NADDA+5)=VEQO
ADDA(NADDA+6)=VEQVO
ADDA(NADDA+7)=VEQMAX
IDA(NADDA+1)=COM17
IDA(NADDA+2)=COM11
IDA(NADDA+3)=COM12

```



```

IDA(NADDA+4)=COMT3
IDA(NADDA+5)=COMT4
IDA(NADDA+6)=COMT5
IDA(NADDA+7)=COMT6
NADDA=NADDA+7
IF(IOP.EQ.0) GO TO 2
IDA(NADDA+1)=COMT8
ADDA(NADDA+1)=FLOAT (IOP)
NADDA=NADDA+1
CONTINUE
2 AL=EXP1/(XO*XD)
XMAXXO=XO*VEQQ
XMAXXVO=XO*VEQVO
XVMAX=VEQMAX*XO
XL=XVMAX
IF(XL.LT.XMAXVO) XL=XMAXVO
IF(IOP.EQ.2) GO TO 210
RORAT=ROMIN/ROINIT
X=1./SQRT (2.*AL)
DX=1/(X-XO)/10.
X=X-1-DX
100 X=X+DX
FX=EXP(-AL*X*X)*{1.-2.*X*(XO-X)*AL}
IF(FX-RORAT) 100,130,110
110 IF(FX-1.005*RORAT) 130,120,120
120 X=X-DX
DX=DX/2.
GO TO 100
130 XIRO=X
ROXI=ROINIT*EXP(-AL*XIRO*XIRO)
IF(ICP.EQ.1) GO TO 210
IFAT=FMIN/FINIT
X=1./SQRT (2.*AL)
IF(EXP(EXPI)-IFAT) 150,200,140
140 DX=(X-XO)/10.
SIGN=1.
GO TO 160
150 DX=X/10.
SIGN=-1.
160 X=X-DX*SIGN
170 X=X+DX*SIGN
FX=EXP(-AL*X*X)*{1.-2.*X*(XO-X)*AL}
IF(FX-IFAT) 170,200,180
180 IF(FX-1.005*IFAT) 200,190,190
190 DX=DX/2.
GO TO 170
200 XI=X

```



```

210 TX1= EXP(-AL*X1T*X1T)*TINIT
CONTINUE
DO 50 J=1,NDZ
DO 50 K=1,NDZ
Z=RZ(NDZ+K)
R=RR(J)
X2=R*R+Z*Z
X=SQRT(X2)
IF(10P.EQ.2) GO TO 1000
IF(X.GT.X1RO) GO TO 3
RO=ROINIT*EXP(-AL*X2)
GO TO 6
3 IF(X.GT.XD) GO TO 4
RO=ROX1*(1.-2.*AL*X1RO*(X-X1RO))
GO TO 6
4 RO=ROMIN
GO TO 6
1000 IF(X.GT.XD) GO TO 4
RG=(ROINIT-ROMIN)*(1.-X/XD)+ROMIN
6 CONTINUE
JRO=J+NDZ*(K-1)
IF(IAV(J).EQ.0) VAR(JRO)=RO
IF(IAV(1).EQ.1) VAR(JRO)=1./RO
COSPHI=R/X
SINPHI=Z/X
IF(X.GT.XVMAX) GO TO 10
V=V0*X/XVMAX
GO TO 12
10 CONTINUE
V=V0
GO TO 13
14 CONTINUE
(Z/R).GT.(ZMAX/RMAX)) GO TO 11
XMAX=ABS(RMAX/COSPHI)
GO TO 12
11 XMAX=ABS(ZMAX/SINPHI)
12 CONTINUE
IF(XMAX.GT.XMAXO) XMAX=XMAXO
V=VG*(X-XMAX)/(XL-XMAX)
IF(X.GT.XMAX) V=0.
13 CONTINUE
JVR=J+NDZ*(K-1+NDZ*(IV(2)-1))
JVZ=J+NDZ*(K-1+NDZ*(IV(4)-1))
VAR(JVR)=V*COSPHI
VAR(JVZ)=V*SINPHI
IF(IV(3).GT.NV) GO TO 20
15 JVP=J+NDZ*(K-1+NDZ*(IV(3)-1))

```

16080
16090
16100
16110
16120
16130
16140
16150
16160
16170
16180
16190
16200
16210
16220
16230
16240
16250
16260
16270
16280
16290
16300
16310
16320
16330
16340
16350
16360
16370
16380
16390
16400
16410
16420
16430
16440
16450
16460
16470
16480
16490
16500
16510
16520
16530
16540
16550


```

20  VAR(JVP)=0.
    CONTINUE
    IF(IOP.EQ.1) GO TO 27
    IF(IOP.EQ.2) GO TO 2000
    IF(X.GT.X1T) GO TO 23
    T=TINIT*EXP(-AL*X2)
    GO TO 26
23  IF(X.GT.X0) GO TO 24
    T=X1*(1.-2.*AL*X1T*(X-X1T))
    GO TO 28
24  T=TMIN
    GO TO 26
27  IF(X.GT.X0) GO TO 24
    T=TINIT
    GO TO 26
2000 IF(X.GT.X0) GO TO 24
    T=(TINIT-TMIN)*(1.-X/X0)+TMIN
26  CONTINUE
    JTI=J+NDR*(K-1+NDZ*(IV(5)-1))
    IF(IAV(5).EQ.0) VAR(JTI)=T
    IF(IAV(5).EQ.1) VAR(JTI)=T*RO
    JTE=J+NDR*(K-1+NDZ*(IV(6)-1))
    IF(IAV(6).EQ.0) VAR(JTE)=T
    IF(IAV(6).EQ.1) VAR(JTE)=T*RO
    IF(IV(7).GT.NV) GO TO 25
    JBR=J+NDR*(K-1+NDZ*(IV(7)-1))
    VAR(JBR)=0.
25  CONTINUE
    IF(IV(8).GT.NV) GO TO 30
    JBP=J+NDR*(K-1+NDZ*(IV(8)-1))
    VAR(JBP)=0.
30  IF(IV(9).GT.NV) GO TO 35
    JBZ=J+NDR*(K-1+NDZ*(IV(9)-1))
    VAR(JBZ)=BZINIT
35  IF(IV(10).GT.NV) GO TO 50
    L=J
    M=K
    JST=L+NDR*(M-1+(IV(10)-1)*NDZ)
    IAVP1=IAV(10)+1
    GO TO (51,52,53), IAVP1
51  Q=0.5*RR(L)*RR(L)
52  GO TO 54
53  Q=0.5
54  GO TO 54
53  Q=0.5*RR(L)
54  GO TO 54
54  VAR(JST)=Q*BZINIT
50  CONTINUE

```


RETURN
END

SUBROUTINE INITS

```

THIS SUBROUTINE USES FILE DATA
THE SUBROUTINE USES INPUT CARD NUMBER 128
REAL*8 CMT1,CMT2,CMT3,CMT4,IDA
COMMON/C3/ DA(175),DADT(25),VAR(3500)
COMMON/C4/IBCR(10),IBCR(10),IBZZ(10),IBSZ(10),ISHK(10),NZP(3),
1 IMPSHK, JSTEP
COMMON/C6/ISTEP,IV(10),IVV(10),IAV(10)
COMMON/C21/RZ(150)
COMMON/C30/ INP, IOUT1, IOUT2, INP2, IOUT4
2 DIMENSION ADDA(40),IDA(40)
3 EQUIVALENCE (DA(9),NDR),(DA(10),NDZ),(DA(12),DR),(DA(13),DZ),
4 (DA(16),NV),(DA(20),NADDA),(DA(24),IBE),(DA(25),RMIN),
(DA(26),ADDA1),(DA(27),RORINT),(DA(28),INIT),
(DA(29),RORIN),(DA(31),BZINIT),(DA(35),8PINIT),
(RZ,RR)
EQUIVALENCE (XSO=,CMT2/, TRINIT=,CMT3/, RORINT=,
1 DATA CMT1/, XSO=,CMT2/, TRINIT=,CMT3/, RORINT=,
2 CMT4/, VLEFT=,
3 INTCON=5
4 READ(INP,5) ICYL,XSO,TRINIT,RORINT,VLEFT
5 FORMAT(4X,I1,I1,I1,0.3)
ADDA(NADDA+1)=XSO
ADDA(NADDA+2)=TRINIT
ADDA(NADDA+3)=RORINT
ADDA(NADDA+4)=VLEFT
IDA(NADDA+1)=CMT1
IDA(NADDA+2)=CMT2
IDA(NADDA+3)=CMT3
IDA(NADDA+4)=CMT4
NADDA=NADDA+4
DO 50 L=1,NDR
X=RR(L)
DO 50 M=1,NDZ
IF(ICYL.NE.0) X=RZ(NDR*M)
JVR=L+NDR*(M-1+NDZ*(IV(2)-1))
VAR(JVR)=0
IF(X.GT.XSO) GO TO 25
IF(ICYL.EQ.0) VAR(JVR)=VLEFT
25 CONTINUE
IF(IV(3).GT.NV) GO TO 46
JVP=L+NDR*(M-1+NDZ*(IV(3)-1))
VAR(JVP)=0.

```

C C


```

46 IF(IV(4).GT.NV) GO TO 47
   JVZ=L+NDR*(M-1+NDZ*(IV(4)-1))
   VAR(JVZ)=0.
   IF(X.GT.XSO) GO TO 47
   IF(CYL.NE.0) VAR(JVZ)=VLEFT
47 CONTINUE
   IF(X.LE.XSO) GO TO 30
   RO=ROINIT
   T=TRINIT
   GO TO 35
30 RO=ROINIT
   T=TINIT
35 CONTINUE
   JRO=L+NDR*(M-1)
   VAR(JRO)=RO
   IF(IAV(6).EQ.1) T=T*RO
   JTI=L+NDR*(M-1+NDZ*(IV(5)-1))
   JTE=L+NDR*(M-1+NDZ*(IV(6)-1))
   VAR(JTI)=T
   VAR(JTE)=T
   IF(IV(7).GT.NV) GO TO 48
   JBR=L+NDR*(M-1+NDZ*(IV(7)-1))
   VAR(JBR)=0.
48 IF(IV(8).GT.NV) GO TO 49
   JBP=L+NDR*(M-1+NDZ*(IV(8)-1))
   VAR(JBP)=0.
   IF(BCRR(8).EQ.2) VAR(JBP)=BPINIT*RR(1)/RR(L)
49 CONTINUE
   IF(IV(9).GT.NV) GO TO 52
   JBZ=L+NDR*(M-1+NDZ*(IV(9)-1))
   VAR(JBZ)=BZINIT
52 IF(IV(10).GT.NV) GO TO 50
   JSI=L+NDR*(M-1+(IV(10)-1)*NDZ)
   VAR(JSI)=0.5*RR(L)*RR(L)*BZINIT
50 CONTINUE
   RETURN
   END

```

17850

 SUBROUTINE MAT2(J,K)

 VARIABLE DEFINITIONS

RO ... DENSITY
 VR,VP,VZ ... VELOCITIES
 TI ... ION TEMPERATURE
 TE ... ELECTRON TEMPERATURE
 BR,BP,BZ ... PLASMA MAGNETIC FIELDS
 BRE,BPE,BZE ... EXTERNAL MAGNETIC FIELDS (IF EXTERNAL MAGNETIC

CCCCCCCC


```

1 18700 (PZP, IIZP), (PPZM, PPRM, DKZZDZ), (PRMZM, CVRMZM(5)),
2 18710 (PRMZP, CVRMZP(5)), (PRPZM, CVRPZM(5)), (PRPZP, CVRPZP(5)),
3 18720 (PPRMZM, DKRRDR), (PPRPZM, PPRMZP, DKRZDR),
4 18730 (RORMZM, CVRMZM), (RORMZP, CVRMZP), (RORPZM, CVRPZM),
5 18740 (RORPZP, CVRPZP)
1 EQUIVALENCE
1 18750 (SI, SB, AP), (SIE, SBE, APE), (SIEP, SBEP, APEP),
2 18760 (SI, SB, SBERM, APRM), (SIEP, SBRP, APRP), (SIZM, SBZM, APSM),
3 18770 (SI, ZP, SBERP, APZP), (SIEP, SBERM, APERM),
4 18780 (SIEP, SBERP, APERP), (SIEZM, SBERZM, APEZM),
5 18790 (SIEZP, SBERZP, APEZP), (SIEPDR, SBERPDR, APEPDR),
6 18800 (SIEPDR, SBERPDR, APEPDR), (SIEPRM, SBERPRM, APEPRM),
7 18810 (SIEPRP, SBERPRP, APEPRP), (SIEPZM, SBERPZM, APEPZM),
8 18820 (SIEPZP, SBERPZP, APEPZP)
9 EQUIVALENCE
1 18830 (T(1), I), (TC(1), K), (TTC(1), KIZM), (TCRM(1), KIRM),
2 18840 (TCRP(1), KIRP), (TCZM(1), KIZM), (TCZP(1), KIZP),
3 18850 (TTC(1), KIZM),
4 18860 (TTC(1), KIZM),
5 18870 (TTC(1), KIZM),
6 18880 (TTC(1), KIZM),
7 18890 (TTC(1), KIZM),
8 18900 (TTC(1), KIZM),
9 18910 (TTC(1), KIZM),
10 18920 (TTC(1), KIZM),
11 18930 (TTC(1), KIZM),
12 18940 (TTC(1), KIZM),
13 18950 (TTC(1), KIZM),
14 18960 (TTC(1), KIZM),
15 18970 (TTC(1), KIZM),
16 18980 (TTC(1), KIZM),
17 18990 (TTC(1), KIZM),
18 19000 (TTC(1), KIZM),
19 19010 (TTC(1), KIZM),
20 19020 (TTC(1), KIZM),
21 19030 (TTC(1), KIZM),
22 19040 (TTC(1), KIZM),
23 19050 (TTC(1), KIZM),
24 19060 (TTC(1), KIZM),
25 19070 (TTC(1), KIZM),
26 19080 (TTC(1), KIZM),
27 19090 (TTC(1), KIZM),
28 19100 (TTC(1), KIZM),
29 19110 (TTC(1), KIZM),
30 19120 (TTC(1), KIZM),
31 19130 (TTC(1), KIZM),
32 19140 (TTC(1), KIZM),
33 19150 (TTC(1), KIZM),
34 19160 (TTC(1), KIZM),
35 19170 (TTC(1), KIZM)

```



```

18 KKRZ(L,M)=TTCD(L,M)*(VVARI(7,M)+EB(1,M))*(VVARI(9,M)+EB(3,M))/BSQT
1(M)
DO 19 M=1,4
19 CKRZ(L,M)=CTD(L,M)*(CVARI(7,M)+CBE(1,M))*(CVARI(9,M)+CBE(3,M))/CBS
1QT(M)
IF(MATAL-GE.4) GO TO 20
DKRRDR(L)=DRMP*KKRR(L,3)-DRPM*KKRR(L,1)+DDRPM*KKRR(L,2)
DKZZDZ(L)=DZMP*KKZZ(L,3)-DZPM*KKZZ(L,1)+DDZPM*KKZZ(L,2)
DKRZDR(L)=DRMP*KKRZ(L,4)-DRPM*KKRZ(L,2)+DDRPM*KKRZ(L,3)
DKRZDZ(L)=DZMP*KKRZ(L,5)-DZPM*KKRZ(L,1)+DDZPM*KKRZ(L,3)
20 CONTINUE
IFLAG=1
NCOM23=85
IF(I*V(6),.EQ.1) NCOM23=87
GO TO 30
25 DO 27 L=1,2
KKR(L)=KT(L)
KZZ(L)=KT(L)
KKRZ(L)=0.
DKRRDR(L)=DKDR(L)
DKZZDZ(L)=DKDZ(L)
DKRZDZ(L)=0.
DKRZDR(L)=0.
27 CONTINUE
IFLAG=0
30 CONTINUE
C TIME LINEARIZED EQUATIONS HAVE FORM A*U+B*DUDX+D*D2UDX2=V
GO TO (131,145,131,145),ISTEP
131 CONTINUE
DXP=DKP
XM=DRM
GO TO 155
145 CONTINUE
DXP=DZP
DXM=DZM
155 CONTINUE
DO 160 L=1,10
V(L)=0.
DO 180 M=1,10
A(L,M)=0.
B(L,M)=0.
D(L,M)=0.
U02=U0*U0
U0R=U0*R
R2=R*R
GO TO (200,300,200,300),ISTEP
200 CONTINUE
IF((NDIM.EQ.1).AND.(JSTEP.EQ.2)) GO TO 300

```

19180
19190
19200
19210
19220
19230
19240
19250
19260
19270
19280
19290
19300
19310
19320
19330
19340
19350
19360
19370
19380
19390
19400
19410
19420
19430
19440
19450
19460
19470
19480
19490
19500
19510
19520
19530
19540
19550
19560
19570
19580
19590
19600
19610
19620
19630
19640


```

C-----
A(1,1)=1./DT+DVZDZ
B(1,1)=VR
A(1,2)=(ICoord+1.)*RO/R
B(1,2)=RO
A(1,4)=DRODZ
V(1)=RO/DT
C-----
A(2,1)=VR*(-1./DT+DVRDR-DVZDZ)+VZ*DVRDZ
A(2,2)=RO*(1./DT-DVRDR-(ICoord+1.)*2.*VR/R-DVZDZ)-VR*DRODR-VZ*DROD
1Z
210 A(2,4)=RO*DVRDZ-VR*DRODZ
V(2)=RO*VZ*DVRDZ-VR*(VR*DRODR+VZ*DRODZ+RO*(ICoord+1.)*VR/R+DVZDZ)
1) IF(IAV(6).EQ.1) GO TO 211
B(2,1)=VR*VR+II+E
B(2,5)=RO
B(2,6)=RO
GO TO 212
211 B(2,1)=VR*VR
B(2,5)=1.
B(2,6)=1.
212 CONTINUE
C-----
V(8)=DVPDZ*(AEPDR+AEP/R)
C-----
225 IF(IV(4).GT.NV) GO TO 240
B(4,1)=VR*VZ
230 A(4,2)=-RO*VZ/R
B(4,2)=-RO*VZ
A(4,4)=RO/DT-VR*DRODR-2.*VZ*DRODZ-RO*VR/R-RO*DVRDR
B(4,4)=RO*VR
V(4)=RO*VR*DVRDZ-VR*(VR*DRODR+VZ*DRODZ+RO*(VR/R+DVRDR))
1) IF(IAV(6).EQ.1) GO TO 231
A(4,1)=VZ/DT+VR*DVRDZ+DIIDZ+DTEDZ
A(4,5)=DRODZ
A(4,6)=A(4,5)
GO TO 232
231 A(4,1)=VZ/DT+VR*DVRDZ
V(4)=V(4)-DPIIDZ-DPEDZ
232 CONTINUE
240 CONTINUE
C-----
IF(IAV(6).EQ.1) GO TO 263
DO 250 I=1,2
I=4+I
IE=11-I
E=3-I

```

19650
19660
19670
19680
19690
19700

19710
19720
19730
19740
19750
19760
19770
19780
19790
19800
19810
19820
19830
19840
19850

19860

19870
19880
19890
19900
19910
19920
19930
19940
19950
19960
19970
19980
19990
20000
20010
20020

20030
20040
20050
20060
20070


```

A(I1,1)=-T(I1)/DT+VZ*DTDZ(I)+GAM2*T(I)*DVZDZ+FEQ*(T(I)-T(E))
B(I1,1)=-VR*T(I)
B(I1,2)=GAM2*RO*T(I)
B(I1,4)=B(I1,2)*(ICoord+1.)/R
A(I1,4)=RO*DTDZ(I)-T(I)*DRODZ
A(I1,1)=RO*(1./DT+GAM2*(DVRDR+(ICoord+1.)*VR/R+DVZDZ)+FEQ)-VR*DRO
1 DR-VZ*DRODZ
D(I1,1)=-KR(I)
B(I1,1)=RO*VR-DKRRDR(I)-(ICoord+1.)*KRR(I)/R-DKRZDZ(I)
A(I1,1E)=-FEQ*RO
AFEQ(I1)=RO*(T(I)-T(E))
V(I1)=RO*(VZ*DTDZ(I)+GAM2*T(I)*(DVRDR+(ICoord+1.)*VR/R+DVZDZ)+2.*F
1 EQ*(T(I)-T(E)))-T(I)*(VR*DRGOR+VZ*DRODZ)
V(I1)=V(I1)+KZZ(I)*DZTDZ2(I)+DTDZ(I)*(KRZ(I)/R+DKZZDZ(I)+DKRZDR(I)
1)
AKT(I)=0.
BKT(I)=0.
CONTINUE
250 DO 260 I=1,2
IF(KRZ(I).EQ.0.) GO TO 260
C CODING FOR MIXED SECOND PARTIAL DERIVATIVES
I=4+I
JTP=J+NDR*(K+NDZ*(IV(I1)-1))
DTDRZP=DRMP*VAR(JTP+1)-DRPM*VAR(JTP-1)+DDRPM*VAR(JTP)
DTDXP(I)=DTDRZP
IF(K.EQ.2) GO TO 255
IU=NAU+IV(I1)+NV*(J-1)
DTRZM=DRMP*(IU+NV)-DRPM*(IU-NV)+DORPM*U(IU)
DIDX(I)=DTRZM
B(I1,1)=B(I1,1)-KRZ(I)/DZM
V(I1)=V(I1)-KRZ(I)*DTRZM/DZM
C ASSUMED DTDZ=0. AT BOUNDARY
255 V(I1)=V(I1)+KRZ(I)*(DTDRZP-DTR(I))/DZP
260 CONTINUE
GO TO 267
C-----
263 CONTINUE
GAM=GAM1+1.
DO 264 I=1,2
I1=4+I
IE=I1-I
E=3-I
B(I1,2)=GAM*P(I)
A(I1,2)=B(I1,2)*(ICoord+1.)/R
A(I1,4)=DPDZ(I)
A(I1,1)=1./DT+GAM*DVZDZ+FEQ
B(I1,1)=VR
A(I1,1E)=-FEQ

```



```

V(I1)=P(I1)/DT
AFEQ(I1)=0.
AKT(I1)=0.
BKT(I1)=0.
CONTINUE
264
C-----
267 CONTINUE
A613=0.
C-----
IF(IV(7).GT.NV) GO TO 275
A(2,9)=-DBRDZ/UO
V(2)=V(2)+BZEP*DBRDZ/UO
A(4,7)=DBRDZ/UO
B(4,9)=-{(BR+BRE)/UO
V(4)=V(4)-BREP*DBRDZ/UO
A613=A(13)+GAM1*DBRDZ*(2.*DBZDR-DBRDZ)/UO2
V(6)=V(6)+GAM1*DBRDZ*(2.*DBRDZ*DBZDR/UO2
B(9,4)=-{(BR+BRE)/R
A(7,2)={BR+BRE}/R
A(7,4)=DBRDZ+DBREDZ
A(7,7)=1./DT+DVZDZ+RES/(UO*R*R)
D(7,7)=-RES/UO
B(7,7)=VR-RES/(UO*R)
A(7,9)=-DV RDZ
B(7,9)=DRES DZ/UO
A713=-DBRDZ /UO
B713=0.
V(7)=BR/DT+DRES DZ*DBRDZ/UO-(BREP-BRE)/DT-VR* BREPDR-DVZDZ*BREP+DVR
1DZ*BZEP
C-----
275 IF(IV(8).GT.NV) GO TO 285
B(2,8)={BP+8PE)/UO
A(2,8)=B(2,8)/R
A(4,8)=DBPDZ/UO
V(4)=V(4)-BPEP*DBPDZ/UO
B(6,8)=-2.*GAM1*RES*(BP/R+DBPDR)/UO2
A(6,8)=B(6,8)/R
A613=A(13)-GAM1*DBPDZ*DBPDZ/UO2
V(6)=V(6)-GAM1*RES*(BP/R+DBPDR)*(BP/R+DBPDR)/UO2
B(8,2)={BP+8PE
A(8,4)=DBPDZ+DBEDZ
A(8,8)=1./DT+DVZDZ+RES/(UO*R*R)
B(8,8)=VR-RES/(UO*R)-DRES DR/UO
D(8,8)=-RES/UO
A813=-DBPDZ /UO
B813=-{(BP/R+DBPDR)/UO
V(8)=V(8)+8P/DT-(8PEP-8PE)/DT-VR* 8PEPDR-OVZDZ*8PEP-DBPDR*DRES DR/UO
1O+DRES DZ*DBPDZ/UO
C-----
285
2800
2810
2820
2830
2840
2850
2860
2870
2880
2890
2900
2910
2920
2930
2940
2950
2960
2970
2980
2990
3000

```



```

C-----285 IF(IV(9).GT.NV) GO TO 295
      B(2,9)=(BZ+BZE)/UO
      B(6,9)=-2.*GAM1*RES*(DBZDR-DBRDZ)/UO2
      V(6)=V(6)-GAM1*RES*DBZDR*DBZDR/UO2
      A(9,2)=(BZ+BZE)/R
      A(9,2)=BZ+BZE
      A(9,4)=DBZDZ+DBZEDZ
      A(9,9)=1./DT
      B(9,9)=VR-RES/(UO*R)-DRESDR/UO
      D(9,9)=-RES/UO
      A913=-D2BZDZ /UO
      B913=-DBZDR/UO+DBRDZ/UO
      V(9)=BZ/DT-(BZEP-BZE)/DT-VR* BZEPDR-DRESDR*DBZDR/UO
      GO TO 400
C-----295 IF(IV(10).GT.NV) GO TO 400
      IAVP1=IAV(10)+1
      GO TO (2950,297,2980),IAVP1
2950 CONTINUE
      UOR2=UO*R*R
      JPQ=D2SIDZ +D2SIDR -DSIDR/R
      B(2,10)={JPQ-(DSIDR+DSIEDR)/R}/UOR2
      D(2,10)={DSIDR+DSIEDR}/UOR2
      V(2)=V(2)- SIEPDR*JPQ/UOR2+(DSIDR+DSIEDR)*(D2SIDR -DSIDR/R)/UOR2
      D(4,10)={DSIDZ+DSIEDZ}/UOR2
      B(4,10)=-D(4,10)/R
      V(4)=V(4)-(DSIDZ+DSIEDZ)*D2SIDZ /UOR2
      IF((MATALI.EQ.1).OR.(MATALI.EQ.3)) GO TO 296
      D(6,10)=-GAM1*2.*RES*JPQ/UOR2/UO
      B(6,10)=-D(6,10)/R
      A613=A613-GAM1*JPQ*JPQ/UOR2/UO
      V(6)=V(6)-GAM1*2.*RES*(D2SIDR -DSIDR/R)*JPQ/UOR2/UO
      GO TO 298
296 V(6)=V(6)+GAM1*RES*JPQ*JPQ/UOR2/UO
298 CONTINUE
      A(10,4)=DSIDZ+DSIEDZ
      A(10,10)=1./DT
      B(10,10)=VR+RES/UO/R
      D(10,10)=-RES/UO
      A1013=-D2SIDZ /UO
      V(10)=SI/DT-(SIEP-SIE)/DT-VR* SIEPDR
      GO TO 400
C-----297 JPQ=D2SBZDZ +D2SBDR +3.*DSBDR/R
      R2UO=R*R/UO
      A(2,10)=2.*R*JPQ/UO
      D(2,10)=R2UO*(2.*(SB+SBE)/R+DSBDR+DSBEDR)
      20980
      20990
      21000
      21010
      21020
      21030
      21040
      21050
      21060
      21070
      21080
      21090
      21100
      21110
      21120
      21130
      21140
      21150
      21160
      21170
      21180
      21190
      21200
      21210
      21220
      21230
      21240
      21250
      21260
      21270
      21280
      21290
      21300
      21310
      21320
      21330
      21340
      21350
      21360
      21370
      21380
      21390
      21400
      21410
      21420

```



```

B(2,10)=0.*R*A(2,10)+3.*D(2,10)/R      21430
V(2)=V(2)-R2U0*JPQ*(2.*SBE/R+DSBEDR)+R2U0*(2.*(SB+SBE)/R+DSBDR+DSB      21440
EDR)*(D2SBDR +3.*DSBDR/R)                21450
D(4,10)=R2U0*(DSBDZ+DSBEDZ)               21460
B(4,10)=3.*D(4,10)/R                     21470
V(4)=V(4)-R2U0*(DSBDZ+DSBEDZ)*D2SBDR      21480
IF((MATALT.EQ.1).OR.(MATALT.EQ.3)) GO TO 299 21490
A613=A613-GAM1*JPQ*JPQ*R2/U02             21500
D(6,10)=-GAM1*2.*RES*JPQ/U02             21510
B(6,10)=3.*D(6,10)/R                     21520
V(6)=V(6)-GAM1*2.*RES*R2*JPQ*(D2SBDR +3.*DSBDR/R)/U02 21530
GO TO 301                                  21540
V(6)=V(6)+GAM1*RES*JPQ*JPQ*R2/U02         21550
CONTINUE                                  21560
A(10,10)=1./DT+VR/R                       21570
A(10,4)=DSBEDZ+DSBDZ                      21580
B(10,10)=VR-3.*RES/U0/R                  21590
D(10,10)=-RES/U0                         21600
A1013=-JPQ/U0                             21610
V(10)=SB/DT-(SBEP-SBE)/DT-VR*(SBEP/R+ SBEPDR)-RES*(D2SBDR +3.*DSBD      21620
R)/R)/U0                                   21630
GO TO 400                                  21640
2980 JPQ=D2APDR +D2APDZ +{(DAPDR-AP/R)/R}    21650
D(2,10)={DAPDR+DAPEDR+(AP+APE)/R}/U0      21660
B(2,10)={D2APDR +D2APDZ +(2.*DAPDR+DAPEDR+APE/R)/R}/U0 21670
A(2,10)={D2APDR +D2APDZ -(2.*AP+APE)/R+DAPEDR/R}/U0/R 21680
V(2)=V(2)-(JPQ-D2APDZ)*(DAPDR+APEP/R)/U0 21690
D(4,10)={DAPDZ+DAPEDZ}/U0                21700
B(4,10)=D(4,10)/R                        21710
A(4,10)=-B(4,10)/R                       21720
V(4)=V(4)-D2APDZ *D(4,10)               21730
GM1U02=GAM1/U0/U0                         21740
IF((MATALT.EQ.1).OR.(MATALT.EQ.3)) GO TO 2981 21750
A613=A613-GM1U02*JPQ*JPQ                 21760
D(6,10)=-GM1U02*2.*RES*JPQ              21770
B(6,10)=D(6,10)/R                       21780
A(6,10)=-B(6,10)/R                      21790
V(6)=V(6)-GM1U02*2.*RES*JPQ*(JPQ-D2APDZ ) 21800
GO TO 2982                                21810
V(6)=V(6)+ GM1U02*RES*JPQ*JPQ           21820
CONTINUE                                  21830
A(10,4)=DAPDZ+DAPEDZ                     21840
A(10,10)=1./DT+(VR+RES/U0/R)/R           21850
B(10,10)=VR-RES/U0/R                    21860
D(10,10)=-RES/U0                        21870
A1013=-D2APDZ /U0                       21880
V(10)=AP/DT-(APEP-AP)/(APEPDR+APEP/R)     21890
2981                                         21900
2982

```


C	GO TO 400	21910
300	A(1,1)=1./DT+(ICORD+1.)*VR/R+DVRDR	21920
	B(1,1)=VZ	21930
	A(1,2)=DRODR	21940
	B(1,4)=RO	21950
	V(1)=RC/DT	21960
C		
	B(2,1)=-VR*VZ	21970
	A(2,2)=RO/DT+RO*DVRDR-VR*DRODR	21980
	B(2,2)=RO*VZ	21990
310	B(2,4)=-RO*VR	22000
	IF(I,AV(6).EQ.1) GO TO 311	22010
	A(2,1)=-VR*(1./DT+DVRDR+(ICORD+1.)*VR/R)+DTEDR+DTIDR	22020
	A(2,5)=DRODR	22030
	A(2,6)=DRODR	22040
	GO TO 312	22050
311	A(2,1)=-VR*(1./DT+DVRDR+(ICORD+1.)*VR/R)	22060
312	V(2)=-DPIDR-DPEDR	22070
C	CONTINUE	22080
	IF(IV(3).GT.NV) GO TO 325	22090
	B(3,10)=(DBPDR+8P/R)/UO	22100
C		
325	IF(IV(4).GT.NV) GO TO 335	22110
	A(4,1)=-VZ/DT-VZ*VR/R-VZ*DVRDR	22120
	A(4,2)=RO*DVZDR-VZ*DRODR	22130
	A(4,4)=RO/DT	22140
	IF(I,AV(6).EQ.1) GO TO 331	22150
	B(4,1)=-VZ*VZ+IE+I	22160
330	B(4,5)=RO	22170
	B(4,6)=RO	22180
	GO TO 332	22190
331	B(4,1)=-VZ*VZ	22200
	B(4,5)=1.	22210
	B(4,6)=1.	22220
332	CONTINUE	22230
335	CONTINUE	22240
C		
	IF(I,AV(6).EQ.1) GO TO 363	22250
	DO 350 I=1,2	22260
	II=4+I	22270
	IE=11-II	22280
	E=3-I	22290
	A(II,1)=T(I)*(-1./DT+GAM2*(DVRDR+(ICORD+1.)*VR/R)+FEQ)-FEQ*T(E)+V	22300
18	*DT*G(I)	22310
	B(II,1)=-T(II)*VZ	22320
	A(II,2)=RO*DTDR(I)-T(I)*DRODR	22330


```

22360
22350
22360
22370
22380
22390
22400
22410
22420
22430
22440
22450
22460
22470
22480
22490
22500
22510
22520
22530
22540
22550
22560
22570
22580
22590
22600
22610
22620
22630
22640
22650
22660
22670
22680
22690
22700
22710
22720
22730
22740
22750
22760
22770

B(I,4)=GAM2*T(I)*RO
A(I,1,1)=RO*(1./DT+FEQ)
D(I,1,1)=KZZ(I)
B(I,1,1)=RO*VZ-DKZZ(I)-KRZ(I)/R-DKRZDR(I)
A(I,1,1)=FEQ*RO
AFEQ(I,1)=RO*(T(I)-T(E))
V(I,1)=RO*(VR*DTDR(I)+2.*FEQ*(T(I)-T(E)))
V(I,1)=V(I)+KRP(I)*(DTOR(I)*(ICORD+1.)/R+D2TOR2(I))+(DKRRDR(I)+DK
1RZDZ(I))*DTDR(I)
AKT(I)=0.
BKT(I)=0.
350 CONTINUE
DO 360 I=1,2
IF(KRZ(I).EQ.0.) GO TO 360
C CODING FOR MIXED SECOND PARTIAL DERIVATIVES
I=4+I
JTRP=J+1+NDR*(K-1+NDZ*(IV(I,1)-1))
DTDZRP=DZMP*VAR(JTRP+NDR)-DZPM*VAR(JTRP-NDR)+DDZPM*VAR(JTRP)
DTDXP(1)=DTDZRP
IF(J.EQ.2) GO TO 355
IU=NAU+IV(I)+NV*(K-1)
DTDZRM=DZMP*(IU+NV)-DZPM*(IU-NV)+DDZPM*(IU)
DTDXM(1)=DTDZPM
B(I,1,1)=B(I,1,1)-KRZ(I)/DRM
V(I,1)=V(I,1)-KRZ(I)*DTDZRM/DRM
C ASSUMED DTDR=0. AT BOUNDARY
355 V(I,1)=V(I)+KRZ(I)*(DTDZRP-DTDZ(I))/DRP
360 CONTINUE
GO TO 367
-----
363 CONTINUE
GAM=GAM1+1.
DO 364 I=1,2
I=4+I
IE=11-I
IE=3-I
B(I,4)=GAM*P(I)
A(I,1,2)=DPDR(I)
A(I,1,1)=1./DT+GAM*((ICORD+1.)*VR/R+DVRDR)+FEQ
B(I,1,1)=VZ
A(I,1,1)=VZ
V(I,1)=P(I)/DT
AFEQ(I)=0.
AKT(I)=0.
BKT(I)=0.
364 CONTINUE
C-----
367 CONTINUE

```



```

C-----
A613=0.
IF(IV(7).GT.NV) GO TO 365
B(2,7)=- (BZ+BZE)/UO
B(4,7)=- (BR+BRE)/UO
A(4,7)=-DBZDR/UO
V(4,7)=BREP*DBZDR/UO
B(6,7)=2.*GAM1*RES*(DBZDR-DBRDZ)/(UO*UO)
V(6,7)=V(6)-GAM1*RES*DBRDZ*DBRDZ/UO2
A(9,7)=-DVZDR
B(9,7)=DRESDR/UO
V(9,7)=DVZDR*BREP
A(7,2)=DBRDR+DBREDR
B(7,2)=-BZ-BZE
B(7,4)=BR+BRE
A(7,7)=1./DT+VR/R
D(7,7)=-RES/UO
B(7,7)=VZ-DRESZ/UO
A713=- (D2BRDR -DBRDR/R+BR/(R*R))/UO
B713=- (DBRDZ+DBZDR)/UO
V(7)=BR/DT-(BREP-BRE)/DT-VZ* BREP-DZ-VR*BREP/R-DRESZ*DBRDZ/UO
C-----

365 IF(IV(8).GT.NV) GO TO 385
A(2,8)=- (BP/R+DBPDR)/UO
V(2,8)=V(2)- (BP/R+DBPDR) *BPEP/UO
B(4,8)=- (BP+BPE)/UO
B(6,8)=-2.*GAM1*RES*DBPDZ/(UO*UO)
A613=A613-GAM1*(BP/R+DBPDR)*(BP/R+DBPDR)/UO2
V(6,8)=V(6)-GAM1*RES*DBPDZ/UO2
370 A(8,2)=DBPDR+DBPEDR
B(8,4)=BP+BPE
A(8,8)=1./DT+DVRDR-DRESDR/(UO*R)
B(8,8)=VZ-DRESZ/UO
D(8,8)=-RES/UO
A813=- (D2BPDR -DBPDR/R+BP/(R*R))/UO
B813=-DBPDZ/UO
V(8)=V(8)+BP/DT-(BPEP-BPE)/DT-VZ* BPEP-DZ-DVRDR*BPEP+DRESDR*DBPDR/UO
10-DBPDZ*DRESZ/UO
C-----

385 IF(IV(9).GT.NV) GO TO 395
A(2,9)=DBZDR/UO
V(2,9)=V(2)-BZEP*DBZDR/UO
A613=A613-GAM1*DBZDR*DBZDR/UO2
A(9,2)=DBZDR+DBZEDR
A(9,9)=1./DT+VR/R+DVRDR
B(9,9)=-RES/UO
D(9,9)=-RES/UO
A913=- (D2BZDR +DBZDR/R)/UO
22780
22790
22800
22810
22820
22830
22840
22850
22860
22870
22880
22890
22900
22910
22920
22930
22940
22950
22960
22970
22980
22990
23000
23010
23020
23030
23040
23050
23060
23070
23080
23090
23100
23110
23120
23130
23140
23150
23160
23170
23180
23190
23200
23210
23220

```



```

      B913=0,
      V(9)=V(9)+BZ/DT-(BZEP-BZE)/DT-VZ* BZEPDZ-VR*BZEP/R-DVRDR*BZEP+DRES
      1DR*DBZDR/UO
      GO TO 400
C-----
395 IF(IV(10).GT.NV) GO TO 400
      IAVP1=IAV(10)+1
      GO TO (3950,397,3980), IAVP1
3950 CONTINUE
      UOR2=UO*R*R
      JPQ=D2SIDZ +D2SIDR -DSIDR/R
      D(2,10)=(DSIDR+DSIEDR)/UOR2
      V(2)=V(2)-(DSIDR+DSIEDR)*(D2SIDR -DSIDR/R)/UOR2
      D(4,10)=(DSIDZ+DSIEDZ)/UOR2
      B(4,10)=JPQ/UOR2
      V(4)=V(4)+(D2SIDZ *(DSIDZ+DSIEDZ)-JPQ* SIEPDZ)/UOR2
      IF((MATALT.EQ.1).OR.(MATALT.EQ.3)) GO TO 396
      D(6,10)=--GAMI*2.*RES*JPQ/UOR2/UO
      V(6)=V(6)+GAMI*RES*((D2SIDR -DSIDR/R)*(D2SIDR -DSIDR/R)-D2SIDZ *D2
      1SIDZ )/UOR2/UO
      GO TO 398
396 V(6)=V(6)+GAMI*RES*JPQ*JPQ/UOR2/UO
398 CONTINUE
      A(10,2)=DSIDR+DSIEDR
      A(10,10)=1./DT
      B(10,10)=VZ
      D(10,10)=--RES/UO
      A(1013)=(-D2SIDR+DSIDR/R)/UO
      V(10)=S1/DT-(SIEP-SIE)/DT-VZ* SIEPDZ
      GO TO 400
C-----
397 JPQ=D2SBDR +D2SBDZ +3.*DSBDR/R
      R2UO=R*R/UO
      D(2,10)=R2UO*(2.*(SB+SBE)/R+DSBEDR+DSBDR)
      V(2)=V(2)-D(2,10)*(D2SBDR +3.*DSBDR/R)
      B(4,10)=R2UO*JPQ
      D(4,10)=R2UO*(DSBDRZ+DSBEDZ)
      V(4)=V(4)+R2UO*(DSBDRZ+DSBEDZ)*D2SBDZ -R2UO*JPQ* SBEPDZ
      IF((MATALT.EQ.1).OR.(MATALT.EQ.3)) GO TO 399
      D(6,10)=--GAMI*2.*RES*R2*JPQ/UO2
      V(6)=V(6)+GAMI*RES*R2*(-D2SBDZ *D2SBDZ +(D2SBDR +3.*DSBDR/R)*(D2SB
      1DR +3.*DSBDR/R))/UO2
      GO TO 401
399 V(6)=V(6)+GAMI*RES*R2*JPQ*JPQ/UO2
401 CONTINUE
      A(10,10)=1./DT
      B(10,10)=VZ
      D(10,10)=--RES/UO

```

23230
 23240
 23250
 23260

 23270
 23280
 23290
 23300
 23310
 23320
 23330
 23340
 23350
 23360
 23370
 23380
 23390
 23400
 23410
 23420
 23430
 23440
 23450
 23460
 23470
 23480
 23490
 23500
 23510

 23520
 23530
 23540
 23550
 23560
 23570
 23580
 23590
 23600
 23610
 23620
 23630
 23640
 23650
 23660
 23670
 23680


```

3980 A(10,2)=(SB+SBE)/R+DSBDR+DSBEDR
      V(10)=SB/DT-(SBEP-SBE)/DT-VZ* SBEPDZ+RES*(D2SBDR +3.*DSBDR/R)/UO
      GO TO 400
      JPQ=D2APDR +D2APDZ +(DAPDR-AP/R)/R
      D(2,10)=(DAPDR+DAPEDR+(AP+APE)/R)/UO
      V(2)=V(2)-D(2,10)*JPQ-D2APDZ
      D(4,10)=(DAPDZ+DAPEDZ)/UO
      B(4,10)=JPQ/UO
      V(4)=V(4)+D(4,10)*D2APDZ -B(4,10)* APEPDZ
      GM1UO2=GAM1/UO/UO
      IF((MATALT.EQ.1).OR.(MATALT.EQ.3)) GO TO 3981
      D(6,10)=GM1UO2*2.*RES*JPQ
      V(6)=V(6)+ GM1UO2*RES*((JPQ-D2APDZ )*(JPQ-D2APDZ )-D2APDZ *D2APDZ
1)
      GO TO 3982
      V(6)=V(6)+ GM1UO2*RES*JPQ*JPQ
3981 CONTINUE
3982 A(10,2)=(AP+APE)/R+DAPDR+DAPEDR
      A(10,10)=1./DT
      B(10,10)=VZ/UG
      D(10,10)=RES/UG
      A1013=-((JPQ-D2APDZ )/UO
      V(10)=AP/DT-(APEP-APE)/DT-VZ* APEPDZ
C-----
400 CONTINUE
420 IF((NDIM.NE.1).OR.(JSTEP.NE.2)) GO TO 440
      DO 430 L=1,10
      DO 430 M=1,10
      DO 430 N=1,10
      B(L,M)=0.
      B510=0.
      B611=0.
      B713=0.
      B813=0.
      B913=0.
430 CONTINUE
440 CONTINUE
C-----
      V(5)=V(5)-A510*W(1)-A512*W(3)-B510*WW(1)
      V(6)=V(6)-A611*W(2)-A612*W(3)-A613*W(4)-B611*WW(2)
      IF(IV(7).GT.NV) GO TO 470
      V(7)=V(7)-A713*W(4)-B713*WW(4)
      IF(IV(8).GT.NV) GO TO 475
      V(8)=V(8)-A813*W(4)-B813*WW(4)
      IF(IV(9).LE.NV) V(9)=V(9)-A913*W(4)-B913*WW(4)
      IF(IV(10).LE.NV) V(10)=V(10)-A1013*W(4)
C-----
      IF(NV.EQ.9) GO TO 479
      DO 478 L=1,NV

```


24140
24150
24160
24170
24180
24190
24200
24210

24220
24230
24240
24250
24260
24270
24280
24290
24300
24310
24320
24330
24340
24350
24360
24370
24380
24390
24400
24410
24420
24430
24440

24450
24460
24470
24480
24490
24500
24510
24520
24530
24540
24550
24560
24570

```

LL=IVV(L)
V(L)=V(LL)
DO 478 M=1,NV
MM=IVV(M)
A(L,M)=A(LL,MM)
B(L,M)=B(LL,MM)
D(L,M)=D(LL,MM)
478 CONTINUE
479 CONTINUE
C-----
C  CHANGE EQUATIONS TO FORM A*U+B*U(+)-C*U(-)=V
DO 480 L=1,NV
DO 480 M=1,NV
MM=IVV(M)
IDXP1=IDDX(MM)+1
GO TO (485,481,482), IDXP1
481 CONTINUE
A(L,M)=A(L,M)+B(L,M)-2.*D(L,M)/DXP)/DXM
HOLD=B(L,M)
B(L,M)=2.*D(L,M)/DXP/(DXP+DXM)
C(L,M)=(HOLD-2.*D(L,M)/(DXP+DXM))/DXM
GO TO 480
482 CONTINUE
A(L,M)=A(L,M)-(B(L,M)+2.*D(L,M)/DXM)/DXP
B(L,M)=B(L,M)+2.*D(L,M)/(DXP+DXM)/DXP
C(L,M)=-2.*D(L,M)/(DXP+DXM)/DXM
GO TO 480
485 CONTINUE
A(L,M)=A(L,M)+((DXP-DXM)*B(L,M)-2.*D(L,M))/(DXP+DXM)
HOLD=B(L,M)
B(L,M)=(HOLD+DXM+2.*D(L,M))/(DXP*(DXM+DXP))
C(L,M)=(HOLD+DXP-2.*D(L,M))/(DXM*(DXM+DXP))
CONTINUE
480 IF(IAV(6).NE.1) RETURN
C-----

```

```

C  INCLUDE THERMAL CONDUCTION TERMS IN PRESSURE EQUATIONS
DO 605 I=1,2
II=IV(4+I)
VV(I)=V(II)
AA(I)=A(II,II)
BB(I)=B(II,II)
CC(I)=C(II,II)
CONTINUE
605 GO TO (610,650,610,650), ISTEP
610 CONTINUE
IF((NDIM.EQ.1).AND.(JSTEP.EQ.2)) GO TO 670
DO 615 I=1,2
II=IV(4+I)
C1=2.*DRMP*(RP+R)*(KKRR(I,3)+KKRR(I,2))/4./DRP/R

```



```

24580 C2=2.*DRPM*(RM+R)*(KKRR(I,1)+KKRR(I,2))/4./DRM/R
24590 C3=2.*DDRPM*KKRR(I,2)
24600 A(I,1)=A(I,1,1)-(-C1-C2+C3*DDRPM)/RO
24610 B(I,1)=B(I,1,1)-(-C1+C3*DRMP)/RORP
24620 C(I,1)=C(I,1,1)+(-C2-C3*DRPM)/RORM
24630 IF(NDIM.EQ.1) GO TO 615
24640 V(I,1)=V(I,1,1)+2.*DZMP*(KKZZ(I,3)+KKZZ(I,2))*(PZP(I)/ROZP-P(I)/RO)/2
24650 1./OZP-DZPM*(KKZZ(I,1)+KKZZ(I,2))*(P(I)/RO-PZM(I)/ROZM)/2./DZM+DDZP
24660 2N*KKZZ(I,2)*(DZMP*PZP(I)/ROZP-DZPM*PZM(I)/ROZM+DDZPM*P(I)/RO)
24670 615 CONTINUE
24680 IF(ISC.NE.1) RETURN
24690 IF(NDIM.EQ.1) RETURN
-----
24700 IF(K.EQ.2) GO TO 622
24710 DO 620 I=1,2
24720 IU=NAU+IV(4+I)+NV*(J-1)
24730 PPZM(I)=U(IU)
24740 PPRMZM(I)=U(IU+NV)
24750 PPRMZM(I)=U(IU+NV)
24760 620 CONTINUE
24770 622 CONTINUE
24780 DO 625 I=1,2
24790 I1=IV(4+I)
24800 C1=DZMP*DRMP*(KKRZ(I,4)+CKRZ(I,2))/2./R
24810 C2=DZMP*DRPM*(KKRZ(I,2)+CKRZ(I,1))/2./R
24820 C3=DZMP*DDRPM*(KKRZ(I,3)+KKRZ(I,5))/2./R
24830 V(I,1)=V(I,1)+C1*(PRPZP(I)/RORPZP-PRP(I)/RORP)
24840 +C2*(PRMZP(I)/RORMZP-PRM(I)/RORM)
24850 +C3*(PZP(I)/ROZP-P(I)/RO)
24860 1RMZP+DDRPM*PZP(I)/ROZP*(KKRZ(I,5)*(DRMP*PRPZP(I)/RORPZP-DRPM*PRMZP(I)/RO
24870 2RM+DDRPM*P(I)/RO)
24880 IF(K.EQ.2) GO TO 625
24890 C IF(K.EQ.2) ASSUME Z DERIVATIVES ARE ZERO AT BOUNDARY
24900 C1=DZPM*DRMP*(KKRZ(I,4)+CKRZ(I,1))*RP/2./R
24910 C2=DZPM*DRPM*(KKRZ(I,2)+CKRZ(I,3))*RM/2./R
24920 C3=DZPM*DDRPM*(KKRZ(I,3)+KKRZ(I,1))/2./R
24930 V(I,1)=V(I,1)+C1*PPRZM(I)/RORPZM+C2*PPRMZM(I)/RORMZM-C3*PPZM(I)/ROZ
24940 1M
24950 A(I,1)=A(I,1,1)-C3/RO
24960 B(I,1)=B(I,1,1)-C1/RORP
24970 C(I,1)=C(I,1,1)-C2/RORM
24980 CONTINUE
24990 624 V(I,1)=V(I,1)-DZPM*KKRZ(I,1)*(DRMP*PRPZM(I)/RORPZM-DRPM*PPRMZM(I)/R
25000 1ORMZM+DDRPM*PZM(I)/ROZM)
25010 A(I,1)=A(I,1,1)-DZPM*DDRPM*KKRZ(I,3)/RO
25020 B(I,1)=B(I,1,1)-DZPM*DRMP*KKRZ(I,3)/RORP
25030 C(I,1)=C(I,1,1)-DZPM*DRPM*KKRZ(I,3)/RORM

```



```

625 CONTINUE
RETURN
C-----
650 CONTINUE
IF(ISO.NE.1) GO TO 670
IF(J.EQ.2) GO TO 660
DO 655 I=1,2
IU=IAU+IV(4,I)+NV*(K-1)
PRP(I)=U(IU)
PRPZP(I)=U(IU+NV)
PPRMZM(I)=U(IU-NV)
655 CONTINUE
660 CONTINUE
DO 665 I=1,2
VI=IV(4,I)
V(II)=V(II)+DRMP*(RP*KKRZ(I,4)*(DZMP*PRPZP(I)/RORPZP-DZPM*PRPZM(I)
1/RORPZM+DZPM*PRP(I)/RORP)-R*KKRZ(I,3)*(DZMP*PZP(I)/ROZP-DZPM*PZM(I)
2I)/ROZPM+DZPM*P(I)/RO)/R
C1=DRMP*PZM*(KKRZ(I,1)+KKRZ(I,4))/2.
C2=DRMP*DZPM*(KKRZ(I,1)+KKRZ(I,4))/2.
C3=DRMP*DZPM*(KKRZ(I,3)+KKRZ(I,4))/2.
V(II)=V(II)+C1*(PRPZP(I)/RORPZP-PZP(I)/ROZP)
1-C2*(PRPZM(I)/RORPZM-PZM(I)/ROZM)
2+C3*(PRP(I)/RORP-P(I)/RO)
GO TO 665
C IF(J.EQ.2) ASSUME R DERIVATIVES ARE ZERO AT BOUNDARY
V(II)=V(II)-DRPM*RM*KKRZ(I,2)*(DZMP*PPRMZP(I)/RORMZP-DZPM*PPRMZM(I)
1)/RORMZM+DZPM*PPRM(I)/RORM)/R
A(II,II)=A(II,II)-DRPM*KKRZ(I,3)*DZPM/RO
B(II,II)=B(II,II)-DRPM*KKRZ(I,3)*DZPM/ROZP
C(II,II)=C(II,II)-DRPM*KKRZ(I,3)*DZPM/ROZM
662 CONTINUE
C1=DRPM*DZPM*(KKRZ(I,5)+KKRZ(I,1))/2.
C2=DRPM*DZPM*(KKRZ(I,1)+KKRZ(I,3))/2.
C3=DRPM*DZPM*(KKRZ(I,1)+KKRZ(I,2))/2.
V(II)=V(II)-(C1*PPRMZP(I)/RORMZP-C2*PPRMZM(I)/RORMZM+C3*PPRM(I)/RO
1RM)
A(II,II)=A(II,II)-C3/RO
B(II,II)=B(II,II)-C1/ROZP
C(II,II)=C(II,II)-C2/ROZM
665 CONTINUE
C-----
670 CONTINUE
DO 675 I=1,2
VI=IV(4,I)
V(II)=V(II)+2.*(DRMP*(RP+R)*(KKRZ(I,3)+KKRZ(I,2))*(PRP(I)/RORP-P(I)
1)/RO)/4.-DRPM*(RM+R)*(KKRZ(I,1)+KKRZ(I,2))*(P(I)/RO-PRM(I)/ROR
2M)/4.-/DRM+DDRP*R*KKRZ(I,2)*(DRMP*PRP(I)/RORP-DRPM*PRM(I)/RORM+DDR

```



```

3PM#P(11)/RO))/R      GO TO 675
IF(NDIM.EQ.1)+KKZZ(1,3)+KKZZ(1,2))/DZP
C1=DZPM#(KKZZ(1,3)+KKZZ(1,2))/DZM
C2=DZPM#(KKZZ(1,3)+KKZZ(1,2))
C3=2.*DDZPM#KKZZ(1,2)
A(11,11)=A(11,11)+(C1+C2-C3*DDZPM)/RO
B(11,11)=B(11,11)-(C1+C3*DDZPM)/ROZP
C(11,11)=C(11,11)+(C2-C3*DDZPM)/ROZM
675 CONTINUE
      RETURN
      END

```

```

25490
25500
25510
25520
25530
25540
25550
25560
25570
25580
25590

```

```
25600
```

```

SUBROUTINE MATRIX
C THIS SUBROUTINE USES FILE DATA.
C THIS SUBROUTINE USES INPUT CARD NUMBERS 6, 7, AND 8.
      IMPLICIT REAL*8 (C)
      REAL CQA,IDA
      COMMON/VC3/ IBCRR(10),IBCR(10),IBCZZ(10),IB CZ(10),ISHK(10),NZP(3),
1
      COMMON/VC6/ ISTEP,IV(10),IVV(10),IAV(10)
      COMMON/VC22/JDDR(10),JDDZ(10)
      COMMON/VC30/ INP,IOUT1,IOUT2,INP2,IOUT4
      DIMENSION IDA(40)
      EQUIVALENCE

```

```
25610
```

```
25620
```

```
25630
```

```
25640
```

```
25650
```

```
25660
```

```
25670
```

```
25680
```

```
25690
```

```
25700
```

```
25710
```

```
25720
```

```
25730
```

```
25740
```

```
25750
```

```
25760
```

```
25770
```

```
25780
```

```
25790
```

```
25800
```

```
25810
```

```
25820
```

```
25830
```

```
25840
```

```
25850
```

```
25860
```

```
25870
```

```
25880
```

```
25890
```

```
25900
```

```
25910
```

```
25920
```

```
25930
```

```
25940
```

```
25950
```

```
25960
```

```
25970
```

```
25980
```

```
25990
```

```
26000
```

```
26010
```

```
26020
```

```
26030
```

```
26040
```

```
26050
```

```
26060
```

```
26070
```

```
26080
```

```
26090
```

```
26100
```

```
26110
```

```
26120
```

```
26130
```

```
26140
```

```
26150
```

```
26160
```

```
26170
```

```
26180
```

```
26190
```

```
26200
```

```
26210
```

```
26220
```

```
26230
```

```
26240
```

```
26250
```

```
26260
```

```
26270
```

```
26280
```

```
26290
```

```
26300
```

```
26310
```

```
26320
```

```
26330
```

```
26340
```

```
26350
```

```
26360
```

```
26370
```

```
26380
```

```
26390
```

```
26400
```

```
26410
```

```
26420
```

```
26430
```

```
26440
```

```
26450
```

```
26460
```

```
26470
```

```
26480
```

```
26490
```

```
26500
```

```
26510
```

```
26520
```

```
26530
```

```
26540
```

```
26550
```

```
26560
```

```
26570
```

```
26580
```

```
26590
```

```
26600
```

```
26610
```

```
26620
```

```
26630
```

```
26640
```

```
26650
```

```
26660
```

```
26670
```

```
26680
```

```
26690
```

```
26700
```

```
26710
```

```
26720
```

```
26730
```

```
26740
```

```
26750
```

```
26760
```

```
26770
```

```
26780
```

```
26790
```

```
26800
```

```
26810
```

```
26820
```

```
26830
```

```
26840
```

```
26850
```

```
26860
```

```
26870
```

```
26880
```

```
26890
```

```
26900
```

```
26910
```

```
26920
```

```
26930
```

```
26940
```

```
26950
```

```
26960
```

```
26970
```

```
26980
```

```
26990
```

```
27000
```

```
27010
```

```
27020
```

```
27030
```

```
27040
```

```
27050
```

```
27060
```

```
27070
```

```
27080
```

```
27090
```

```
27100
```

```
27110
```

```
27120
```

```
27130
```

```
27140
```

```
27150
```

```
27160
```

```
27170
```

```
27180
```

```
27190
```

```
27200
```

```
27210
```

```
27220
```

```
27230
```

```
27240
```

```
27250
```

```
27260
```

```
27270
```

```
27280
```

```
27290
```

```
27300
```

```
27310
```

```
27320
```

```
27330
```

```
27340
```

```
27350
```

```
27360
```

```
27370
```

```
27380
```

```
27390
```

```
27400
```

```
27410
```

```
27420
```

```
27430
```

```
27440
```

```
27450
```

```
27460
```

```
27470
```

```
27480
```

```
27490
```

```
27500
```

```
27510
```

```
27520
```

```
27530
```

```
27540
```

```
27550
```

```
27560
```

```
27570
```

```
27580
```

```
27590
```

```
27600
```

```
27610
```

```
27620
```

```
27630
```

```
27640
```

```
27650
```

```
27660
```

```
27670
```

```
27680
```

```
27690
```

```
27700
```

```
27710
```

```
27720
```

```
27730
```

```
27740
```

```
27750
```

```
27760
```

```
27770
```

```
27780
```

```
27790
```

```
27800
```

```
27810
```

```
27820
```

```
27830
```

```
27840
```

```
27850
```

```
27860
```

```
27870
```

```
27880
```

```
27890
```

```
27900
```

```
27910
```

```
27920
```

```
27930
```

```
27940
```

```
27950
```

```
27960
```

```
27970
```

```
27980
```

```
27990
```

```
28000
```

```
28010
```

```
28020
```

```
28030
```

```
28040
```

```
28050
```

```
28060
```

```
28070
```

```
28080
```

```
28090
```

```
28100
```

```
28110
```

```
28120
```

```
28130
```

```
28140
```

```
28150
```

```
28160
```

```
28170
```

```
28180
```

```
28190
```

```
28200
```

```
28210
```

```
28220
```

```
28230
```

```
28240
```



```

IOT=49
READ(INP,2) IDVDX,MMAT,MATALT,ISO,DT,CQA,(NZP(L),L=1,3),JJDDR,
      1 JDDZ,NICDRV
2 FORMAT(4I1,6X,2E10.3,3I5,2I10,I5)
  IF(MMAT.EQ.0) MMAT=1
  IF(IFROM.NE.0) GO TO 40
  READ(INP,3) JVER,JAV,INTCON,IBE,ROINIT,TINIT,ROMIN,TRO
3 FORMAT(12,3I1,5X,4E10.3)
4 FORMAT(5X,7E10.3)
  IF(JAV.EQ.0) JAV=1
  MAT=MMAT
  IDA(1)=COMT1
  IDA(2)=COMT2
  IDA(3)=COMT3
  IDA(4)=COMT4
  IDA(5)=COMT5
  GO TO(101,102,103,104,105,106,107), JAV
101 IIAV=0
  GO TO 110
102 IIAV=1000110000
  GO TO 110
  C U=(SV=1./RO.....PI=TI*RO.....PE=TE*RO.....)
103 IIAV=1
  GO TO 110
  C U=(.....SB=SI/R2=AP/R)
104 IIAV=110000
  GO TO 110
  C U=(.....PI=TI*RO.....PE=TE*RO.....)
105 IIAV=2
  GO TO 110
  C U=(.....AP)
106 IIAV=110002
  GO TO 110
  C U=(.....PI=TI*RO,PE=TE*RO.....AP)
107 IIAV=110001
  GO TO 110
  C U=(.....PI=TI*RO,PE=TE*RO.....SB=SI/R2=AP/R)
110 ICOORD=0
  JVP=JVER$+1
  JVM=JVER$-10
  GO TO(109,20,30,33,39,50,60,70,80,90,100,109),JVP
109 GO TO(111,120,130,140),JVM
10 IIV=1234567890
  IIVV=1234567890
  NDIM=2
  NV=9
  GO TO 34

```

```

25930
25940
25950
25960
25970
25980
25990
26000
26010
26020
26030
26040
26050
26060
26070
26080
26090
26100
26110
26120
26130
26140
26150
26160
26170
26180
26190
26200
26210
26220
26230
26240
26250
26260
26270
26280
26290
26300
26310
26320
26330
26340

```



```

C 20 IIV=1283456970
   U=(RO,VR,VZ,II,TE,BR,BZ)
   IIVV=1245679380
   NDIM=2
   NV=7
   NADDA=9
   IDA(9)=COMT6
   GO TO 35
C 20 U=(RO,VR,II,TE,BZ)
   30 IIV=1267348950
   IIVV=1256934780
   NDZ=1
   NDIM=1
   NV=5
   NADDA=8
   GO TO 35
C 33 IIV=1278349560
   U=(RO,VR,II,TE,BP,BZ)
   IIVV=1256893470
   NDZ=1
   NDIM=1
   NV=6
   GO TO 34
C 111 ICOORD=1
   SELECTS RECTANGULAR COORDINATES
   GO TO 39
C 100 ICOORD=1
   SELECTS SPHERICAL COORDINATES
C 39 IIV=1256347890
   U=(RO,VR,II,TE)
   IIVV=1256347890
   NDIM=1
   NDZ=1
   NV=4
   NADDA=5
   GO TO 40
C 50 IIV=1234569708
   U=(RO,VR,VZ,II,TE,BP,SI)
   IIVV=1234568079
   NDIM=2
   NV=8
   GO TO 34
C 60 IIV=1273458906
   U=(RO,VR,VZ,II,TE,SI)
   IIVV=1245603789
   NDIM=2
   NV=6
   NADDA=9

```

```

26350
26360
26370
26380
26390
26400
26410
26420
26430
26440
26450
26460
26470
26480
26490
26500
26510
26520
26530
26540
26550
26560
26570
26580
26590
26600
26610
26620
26630
26640
26650
26660
26670
26680
26690
26700
26710
26720
26730
26740

```



```

IDA(9)=COMT6
GO TO 35
70 IIV=1278349506
C U=(RO,VR,II,IE,BP,SI)
IIVV=1256803479
NDIM=1
NDZ=1
NV=6
GO TO 34
80 IIV=1267348905
C U=(RO,VR,II,IE,SI)
IIVV=1256034789
NDIM=1
NDZ=1
NV=5
NADDA=8
GO TO 35
90 IIV=1263457890
C U=(RO,VR,VZ,II,IE)
IIVV=1245637890
NDIM=2
NV=5
NADDA=5
GO TO 40
120 ICORD=-1
C SELECTS RECTANGULAR COORDINATES
GO TO 140
130 ICORD=1
C SELECTS SPHERICAL COORDINATES
140 IIV=1245637890
C (RO,VR,IE)
IIVV=1263457890
NDIM=1
NDZ=1
NV=3
NADDA=5
GO TO 40
34 NADDA=12
IDA(9)=COMT6
IDA(10)=COMT7
IDA(11)=COMT8
IDA(12)=COMT9
35 CONTINUE
IDA(6)= COMT10
IDA(7)=COMT11
IDA(8)=COMT12
IF(1BCR{9).NE.2) GO TO 36
IDA(7)=COMT13

```



```

36 IDA(8)=COMT14
37 CONTINUE
38 READ(INP,4) BZINIT,BZMAX,BZF,BRAT,BPINIT,BPMAZ,BPF
40 DO 45 L=1,10
41 JDDR(11-L)=JDDR/(10**(L-1))-10*(JDDR/(10**L))
42 JDDZ(11-L)=JDDZ/(10**(L-1))-10*(JDDZ/(10**L))
43 IF(IV(11-L)=IIV/(10**(L-1))-10*(IIV/(10**L)))
44 IF(IV(11-L)=EQ.0) IIV(11-L)=10
45 IF(IV(11-L)=IIVV/(10**(L-1))-10*(IIVV/(10**L)))
46 IF(IVV(11-L)=EQ.0) IIVV(11-L)=10
47 IAV(11-L)=IAV/(10**(L-1))-10*(IAV/(10**L))
48 CONTINUE
49 IF(IFROM.NE.0) GO TO 55
50 IF((IBCR(11).EQ.2).OR.(IBCR(11).EQ.3).OR.(NDIM.EQ.2).AND.(IBZ(11).
51 EQ.2))) IDA(5)=COMT15
52 CONTINUE
53 RETURN
54 END

```

C C

```

SUBROUTINE MESH
THIS SUBROUTINE USES FILE DATA
THIS SUBROUTINE USES INPUT CARD NUMBER 11A OR 11B.
REAL*8 COMT1,COMT2,COMT3,COMT4,IDA
COMMON/C3/DA(175),DADT(25),VAR(3500)
COMMON/C2/RZ(150)
COMMON/C30/INP,IOUT1,IOUT2,INP2,IOUT4
DIMENSION ADDA(40),IDA(40)
EQUIVALENCE (DA(1),NDIM),(DA(9),NDR),(DA(10),NDZ),(DA(12),DRMIN),
1 (DA(13),DZMIN),(DA(20),NADDA),(DA(25),RMIN),
2 (DA(26),ADDA(1)),(DA(56),NREG),(DA(61),Z1),
3 (DA(62),NDZ1),(DA(63),R1),(DA(64),NDR1),(DA(65),ZMAX),
4 (DA(66),RMAX),(DA(71),IDAL1))
EQUIVALENCE (RR,RZ)
DATA COMT1/,DZ RAT=1/, COMT2/,1DR RAT=1/, COMT3/, DZ RAT=1/,
1 COMT4/, DZ RAT=1/
RZ(NDR+NDZ)=0
2 READ(INP,10) NREG,RMIN,RMAX,ZMAX,RRAT,ZRAT
3 FORMAT(4X,11,5E10,3)
4 IF(NREG.EQ.0) NREG=1
5 IF(RRAT.EQ.0) RRAT=1
6 IF(ZRAT.EQ.0) ZRAT=1
7 IF(NREG.GT.1) READ(INP,20) NDR1,R1,RRAT1,NDZ1,Z1,ZRAT1
8 FORMAT(5X,2(15,2E10,3))
9 IF(NREG.GT.1) GO TO 30
10 R1=RMAX
11 NDR1=NDR

```



```

30 IF (RRAT.EQ.1.) GO TO 35
   ADDA(NADDA+1)=RRAT
   IDA(NADDA+1)=COMT1
   NADDA=NADDA+1
35 CONTINUE
   IF (RRAT.EQ.1.) TERM=NDRI-1.
   IF (RRAT.NE.1.) TERM=(RRAT** (NDRI-1)-1.)/(RRAT-1.)
   DR1=(R1-RMIN)/(TERM-0.5)
   DRMIN=DK1
   DO 40 L=1, NDRI
   IF (RRAT.NE.1.) FACTOR=(RRAT** (L-1)-1.)/(RRAT-1.)
   IF (RRAT.EQ.1.) FACTOR=L-1.
   RR(L)=RMIN+DR1*(FACTOR-0.5)
   IF (NREG.LE.1) GO TO 70
   IF (RRAT1.EQ.1.) GO TO 50
   ADDA(NADDA+1)=RRAT1
   IDA(NADDA+1)=COMT2
   NADDA=NADDA+1
50 CONTINUE
   NP=NDR-NDRI
   IF (RRAT1.EQ.1.) TERM=NP
   IF (RRAT1.NE.1.) TERM=(RRAT1**NP-1.)/(RRAT1-1.)
   DR2=(RMAX-R1)/TERM
   DO 60 L=1, NP
   IF (RRAT1.NE.1.) FACTOR=(RRAT1** (L-1)-1.)/(RRAT1-1.)
   IF (RRAT1.EQ.1.) FACTOR=L
   RR(NDRI+L)=R1+DR2*FACTOR
   IF (DRI.GT.DR2) DRMIN=DR2
70 IF (NDIM.EQ.1) GO TO 130
   IF (NREG.GT.1) GO TO 80
   Z1=ZMAX
   NDZ1=NDZ
80 IF (ZRAT.EQ.1) GO TO 90
   ADDA(NADDA+1)=ZRAT
   IDA(NADDA+1)=COMT3
   NADDA=NADDA+1
90 IF (ZRAT.EQ.1.) TERM=NDZ1-1.
   IF (ZRAT.NE.1.) TERM=(ZRAT** (NDZ1-1)-1.)/(ZRAT-1.)
   IF (NREG.LE.1) DZ1=Z1/(TERM-1.)
   IF (NREG.GT.1) DZ1=Z1/(TERM-0.5)
   DZMIN=DZ1
   DO 100 L=1, NDZ1
   IF (ZRAT.NE.1.) FACTOR=(ZRAT** (L-1)-1.)/(ZRAT-1.)
   IF (ZRAT.EQ.1.) FACTOR=L-1.
   RZ(NDZ+L)=DZ1*(FACTOR-0.5)
   IF (NREG.LE.1) GO TO 130
   IF (ZRAT1.EQ.1.) GO TO 110
   ADDA(NADDA+1)=ZRAT1

```

27610
27620
27630
27640
27650
27660
27670
27680
27690
27700
27710
27720
27730
27740
27750
27760
27770
27780
27790
27800
27810
27820
27830
27840
27850
27860
27870
27880
27890
27900
27910
27920
27930
27940
27950
27960
27970
27980
27990
28000
28010
28020
28030
28040
28050
28060
28070
28080


```

5      ,DBREDR,DBPEDR,DBZEDR,DSIEDR,DBREDZ,DBPEDZ,DBZEDZ,
6      DSIEDZ,8REPDR,BPEPDR,DZEPDR,SIEPDR,DREPDR,DPEPDZ,
7      BZEPDZ,SIEPDZ
      COMMON/C13/IDDX(10)
      COMMON/C22/JDDR(10)
      DIMENSION VARI(10),VARIRM(10),VARIRP(10),VARIZP(10),
1      DVDR(10),DVDZ(10),D2VDR2(10),D2VDZ2(10),TCZM(4),TCRM(4),
2      TC(4),TCRP(4),TCZP(4),DTCDR(4),DTCZDZ(4),TTC(4,5),
3      D(10,10)
      EQUIVALENCE (D(1),C(1))
      EQUIVALENCE (DA(1),NDIM), (DA(3),MAT), (DA(9),NDR), (DA(10),NDZ),
      EQUIVALENCE (DA(11),DI), (DA(16),NV), (DA(29),RMIN),
1      (DA(52),MATAI), (DA(55),IDVDX)
      EQUIVALENCE (DA(15),MMAT)
      EQUIVALENCE (VARIRM(1),ROZM), (VARIRP(1),RORM), (VARI(1),RO),
2      (VARIRP(1),RORP), (VARIZP(1),ROZP), (DVDR(1),DRODR),
3      (DVDZ(1),DRODZ), (D2VDR2(1),D2RODR), (D2VDZ2(1),D2DODZ),
4      (TCZM(1),KIZM), (TCRM(1),KIRM), (TC(1),KI),
5      (TCRP(1),KIRP), (TCZP(1),KIZP), (DTCDR(1),DKIDR),
6      (DTCZDZ(1),DKIDZ)
      EQUIVALENCE (TTC(1),KIZM)
      IF(MODE.EQ.2) GO TO 2000
      JDRL=JDDR(1)
      GO TO(31,31,31,31,31,35,36,25,38,25,40),L
31      VAD=VR
      GO TO 37
35      CONTINUE
      IF(IAV(5).EQ.1) GO TO 351
      VAD=VR-(KI/R+DKIDR)/RO
      GO TO 37
351      VAD=VR
      GO TO 37
36      CONTINUE
      IF(IAV(6).EQ.1) GO TO 361
      VAD=VR-(KE/R+DKEDR)/RO
      GO TO 37
361      VAD=VR
      GO TO 37
38      VAD=VR-(RES/R+DRESDR)/UO
      GO TO 37
40      GO TO 25
37      IF(VAD) 100,25,20
20      CONTINUE
      IF(D2VDR2(L)) 22,25,21
21      GO TO (21,21,21,21,25),JDRL
211      IF(DVDR(L)) 25,25,25
213      IF(DVDR(L)-VARIRM(L)) 211,25,25
22      GO TO(221,222,222,25,221),JDRL

```


221	IF(DVDR(L)) 23,25,25	29030
222	IF(VARI(L)-VARI(M(L)) 25,25,221	29040
223	DVDR(L)=(VARI(L)-VARI(M(L)))/DRM	29050
C	JDRL=1 ONESIDED AT EXISTING AND POTENTIAL MAXIMAS, MINIMAS	
C	JDRL=2 ONESIDED AT EXISTING ONLY MAXIMAS	
C	JDRL=3 ONESIDED AT EXISTING ONLY MAXIMA, MINIMAS	
24	GO TO (24,25,24,25), ISTEP	29060
	IDDX(L)=1	29070
	GO TO 25	29080
100	CONTINUE	29090
101	IF(D2VDR2(L)) 101,25,102	29100
1011	GO TO (1011,1012,1012,25,1011), JDR1	29110
1012	IF(DVDR(L)) 25,25,103	29120
1021	IF(VARI(L)-VARI(M(L)) 25,25,1011	29130
1022	GO TO (1021,1021,1022,1021,25), JDR1	29140
1023	IF(DVDR(L)) 103,25,25	29150
104	IF(VARI(L)-VARI(M(L)) 1021,25,25	29160
105	DVDR(L)=(VARI(M(L)-VARI(L)))/DRP	29170
	GO TO (104,25,104,25), ISTEP	29180
	IDDX(L)=2	29190
	CONTINUE	29200
2000	RETURN	29210
41	GO TO (41,41,41,41,45,45,30,48,30,41), L	29220
45	VAD=VZ	29230
	GO TO 47	29240
	CONTINUE	29250
	IF(IAV(5).EQ.1) GO TO 41	29260
	VAD=VZ-DKIDZ/RO	29270
	GO TO 47	29280
46	CONTINUE	29290
	IF(IAV(6).EQ.1) GO TO 41	29300
	VAD=VZ-DKEDZ/RO	29310
	GO TO 47	29320
48	VAD=VZ-DRESOZ/UO	29330
	GO TO 47	29340
47	IF(VAD) 200,30,49	29350
49	CONTINUE	29360
	IF(D2VDZ2(L)) 27,30,26	29370
26	IF(DVDZ(L)) 30,30,28	29380
27	CONTINUE	29390
	JDZL=JDZ(L)	29400
	GO TO (271,272), JDZL	29410
270	CONTINUE	29420
271	IF(DVDZ(L)) 28,30,30	29430
272	IF(VARI(L)-VARI(M(L)) 30,30,271	29440
28	DVDZ(L)=(VARI(L)-VARI(M(L)))/DZM	29450
	GO TO (30,29,30,29), ISTEP	29460
		29470


```

29 IDDX(L)=1
GO TO 30
200 CONTINUE
201 IF(DVDZ(L)) 201,30,202
202 IF(DVDZ(L)) 30,30,203
203 IF(DVDZ(L)) 203,30,204
204 DVDZ(L)=(VAR1ZP(L)-VAR1(L))/DZP
GO TO (30,204,30,204),ISTEP
204 IDDX(L)=2
200 CONTINUE
203 RETURN
END

```

```

SUBROUTINE OUTPT1
SUBROUTINE USES FILE MHDOUT.

```

```

WRITES PROFILES IN THE Z PLANE

```

```

INTEGER ZERO,PLUS,DASH,EYE,EQUAL,STAR,BLANK,PRNCHR
REAL*8 STMT,COMT1,COMT2,COMT3,COMT4
COMMON/C1/DUM1(12),EMASS,PMASS
COMMON/C3/ DA(175),DADT(25),VAR(3500)
COMMON/C4/ IBCR(10),IBCR(10),IBCCZ(10),IBSK(10),NZP(3),
1 COMMON/C6/ IBSHK,J,STEP
COMMON/C21/RZ(150)
COMMON/C30/ INP,IOUT1,IOUT2,INP2,IOUT4
DIMENSION PRNCHR(104)
DIMENSION R(10)
DIMENSION STMT(4)
EQUIVALENCE (DA(1),NDIM),(DA(5),IDATE),(DA(9),NDR),(DA(10),NDZ),
1 (DA(12),DR),(DA(13),DZ),(DA(16),NV),(DA(19),IFYL),
2 (DA(23),ITAPE),(DA(24),IBE),(DA(25),RMIN),
3 (DA(56),NREG),(DA(65),ZMAX),(DA(66),RMAX),
EQUIVALENCE (DADT(1),TIME),(DADT(2),NNDT),(DADT(5),MMAT)
DATA ZERO/0/,PLUS/+'/,DASH/'-'/,EYE/'/'/,EQUAL/'='/,STAR/'*'/,
1 BLANK/' ',
2 DATA COMT1/' ',COMT2 /'. PLASMA'/,COMT3 /' COMPONE'/',
3 COMT4 /'NT ONLY.'/
NPZ=3
NPCHAR=104
ROTONI=10.**((-6)/(EMASS+PMASS))
TIMEV=(10.**(19))*((EMASS+PMASS)/1.602
1 IF(NDIM.EQ.1) NPZ=1
2 IF(NDIM.EQ.1) NZP(1)=1
3 DO 17 J=1,4

```

CCCC

29480
29490
29500
29510
29520
29530
29540
29550
29560
29570
29580
29590

29600

29610
29620
29630
29640
29650
29660
29670
29680
29690
29700
29710
29720
29730
29740
29750
29760
29770
29780
29790
29800
29810
29820
29830
29840
29850
29860
29870
29880
29890


```

17 STMT(J)=COMT1 GO TO 18
   IF(1BE.EQ.0) GO TO 18
   STMT(1)=COMT2
   STMT(2)=COMT3
   STMT(3)=COMT4
18 CONTINUE
   DO 280 J=1,NV
     WRITE(IOUT2,
10 FORMAT('TIME=',I4,3X,'TIME STEP=',I4,3X,'FILE NUMBER=',I3,'-',
A12,2X,
11 I2,3X,A8,4X,'ORIGIN AT R=',E12.6,5X,'OVERALL RADIUS=',E12.6)
     IF(NDIM.EQ.1) GO TO 15
     WRITE(IOUT2,
11 FORMAT('X,AL', 'AT Z=',I3,3X,AL, ' AT Z=',I3,3X,AL, ' AT Z=',I3,63X,
10 OVERALL LENGTH=',E13.6)
15 CONTINUE
     IVAR=IVV(J)
21 CONTINUE
     GO TO (20,30,36,40,50,60,70,76,80,86),IVAR
20 CONTINUE
     IF(IAV(1).EQ.0) WRITE(IOUT2,25) ROTONI
25 FORMAT('IX,DENSITY(KG/M3, TO CONVERT TO G/CM3, MULTIPLY BY 10.E-
13. TO CONVERT TO ION NUMBER/CM3, MULTIPLY BY ',E10.3,')')
     GO TO 90
30 WRITE(IOUT2,35)
35 FORMAT('IX, 64HRADIAL VELOCITY(M/SEC. TO CONVERT TO CM/SEC, MULTI
PLY BY 10.**2)')
     GO TO 90
36 WRITE(IOUT2,37)
37 FORMAT('IX,67HAZIMUTHAL VELOCITY(M/SEC. TO CONVERT TO CM/SEC, MUL
TIPLY BY 10.**2)')
     GO TO 90
40 WRITE(IOUT2,45)
45 FORMAT('IX,63HAXIAL VELOCITY(M/SEC. TO CONVERT TO CM/SEC, MULTIPL
Y BY 10.**2)')
     GO TO 90
50 CONTINUE
     IF(IAV(5).EQ.0) WRITE(IOUT2,55) TTOEV
     IF(IAV(15).EQ.1) WRITE(IOUT2,56) TTOEV
55 FORMAT('IX, ION TEMPERATURE(M2/SEC2. TO CONVERT TO ELECTRON VOLTS
1, MULTIPLY BY ',E10.3,')')
56 FORMAT('IX, ION PRESSURE(N/M2. TO CONVERT TO EV, MULTIPLY BY ',E1
10.3, AND DIVIDE BY DENSITY.))
     GO TO 90
60 CONTINUE
     IF(IAV(6).EQ.0) WRITE(IOUT2,65) TTOEV
     IF(IAV(16).EQ.1) WRITE(IOUT2,66) TTOEV
65 FORMAT('IX, ELECTRON TEMPERATURE(M2/SEC2. TO CONVERT TO ELECTRON

```

29900
29910
29920
29930
29940
29950
29960
29970
29980
29990
30000
30010
30020
30030
30040
30050
30060
30070
30080
30090
30100
30110
30120
30130
30140
30150
30160
30170
30180
30190
30200
30210
30220
30230
30240
30250
30260
30270
30280
30290
30300
30310
30320
30330
30340
30350
30360
30370


```

30380
30390
30400
30410
30420
30430
30440
30450
30460
30470
30480
30490
30500
30510
30520
30530
30540
30550
30560
30570
30580
30590
30600
30610
30620
30630
30640
30650
30660
30670
30680
30690
30700
30710
30720
30730
30740
30750
30760
30770
30780
30790
30800
30810
30820
30830
30840
30850

1VOLTS, MULTIPLY BY ,E10.3,.)
66 FORMAT(/,1X,'ELECTRON PRESSURE(NT/M2, TO CONVERT TO EV, MULTIPLY B
   LY ,E10.3, AND DIVIDE BY DENSITY.);)
   GO TO 90
70 WRITE(IOUT2 ,75) (STMT(K),K=1,4)
75 FORMAT(/,1X,71RADIAL MAGNETIC FIELD(WB/M**2. TO CONVERT TO GAUSS,
   1 MULTIPLY BY 10.**4),4A8)
   GO TO 90
76 WRITE(IOUT2 ,77) (STMT(K),K=1,4)
77 FORMAT(/,1X,74HAZIMUTHAL MAGNETIC FIELD(WB/M**2. TO CONVERT TO GAU
   1SS, MULTIPLY BY 10.**4),4A8)
   GO TO 90
80 WRITE(IOUT2 ,85) (STMT(K),K=1,4)
85 FORMAT(/,1X,70HAZIAL MAGNETIC FIELD(WB/M**2. TO CONVERT TO GAUSS,
   1 MULTIPLY BY 10.**4),4A8)
   GO TO 90
86 IF(IAV(10),EQ.0) WRITE(IOUT2 ,87) (STMT(K),K=1,4)
   IF(IAV(10),EQ.1) WRITE(IOUT2 ,88) (STMT(K),K=1,4)
   IF(IAV(10),EQ.2) WRITE(IOUT2 ,89) (STMT(K),K=1,4)
88 FORMAT(/,1X,'STREAM FUNCTION(AZIMUTHAL VECTOR POTENTIAL(WB/M)/RADI
   1US),4A8)
89 FORMAT(/,1X,'AZIMUTHAL VECTOR POTENTIAL(WB/M)',4A8)
87 FORMAT(/,1X,57HSTREAM FUNCTION(RADIUS*AZIMUTHAL VECTOR POTENTIAL..
   1WB/M),4A8)
90 CONTINUE
   AMAX=VAR(2+NDR*(NZP(1)-1+(J-1)*NDZ))
   AMIN=AMAX
   DO 100 K=1,NPZ
     DO 100 L=2,NDR
       A=VAR(L+NDR*(NZP(K)-1+(J-1)*NDZ))
       IF(A.GE.AMIN) GO TO 95
       AMIN=A
     GO TO 100
95 IF(A.LE.AMAX) GO TO 100
   AMAX=A
100 CONTINUE
   IF(AMAX.NE.AMIN) GO TO 110
   WRITE(IOUT2 ,105) AMAX
105 FORMAT(/,20X,'UNIFORM',E15.7)
   GO TO 280
110 CONTINUE
   IF(AMIN.LE.0.) GO TO 115
   AMIN=0.
   GO TO 120
115 IF(AMAX.GE.0.) GO TO 120
   AMAX=0.
120 CONTINUE
   DX=(RR(NDR)-(RR(1)+RR(2))/2.)/100.

```



```

DY=(AMAX-AMIN)/25.
I2=INT (AMAX/DY+0.499999)+1
DO 250 K=1,26
DO 130 L=1,NPCHAR
  PRNCHR(L)=BLANK
  PRNCHR(2)=EYE
  IF(K.NE.I2) GO TO 150
DO 140 L=1,102
  PRNCHR(L)=DASH
130 AP=AMAX-(K-1.5)*DY
  AN=AMAX-(K-0.5)*DY
DO 190 L=1,NPZ
  LL=NDR*(INTZP(L)-1+(J-1)*NDZ)
DO 190 M=2,NDR
  A=VAR(M,LL)
  IF((A-GE-AP).OR.(A.LT-AN)) GO TO 190
  MM=INT (-(RR(M)-(RR(1)+RR(2))/2.)/DX+0.499999)+2
  GO TO (160,170,180),L
  PRNCHR(MM)=PLUS
160 GO TO 190
170 PRNCHR(MM)=EQUAL
  GO TO 190
180 PRNCHR(MM)=STAR
190 CONTINUE
  IF(K.NE.1) GO TO 210
  PRNCHR(1)=DASH
  WRITE(IOUT2,200) AMAX,(PRNCHR(L),L=1,NPCHAR)
  FORMAT(14X,E14.5,104A1)
  GO TO 250
210 CONTINUE
  IF(K.NE.26) GO TO 220
  PRNCHR(1)=DASH
  WRITE(IOUT2,200) AMIN,(PRNCHR(L),L=1,NPCHAR)
  GO TO 250
220 IF(K.NE.I2) GO TO 240
  WRITE(IOUT2,220) ZERO,(PRNCHR(L),L=1,NPCHAR)
  FORMAT(27X,105A1)
230 GO TO 250
240 WRITE(IOUT2,230) BLANK,(PRNCHR(L),L=1,NPCHAR)
250 CONTINUE
  DO 270 K=1,NDZ
  KKK=NDR*(K-1+(J-1)*NDZ)+1
  KKK=KK+NDR-1
  IF(NDIM.EQ.2) WRITE(IOUT2,260) K
260 FORMAT(14X,2 COORDINATE,14)
265 WRITE(IOUT2,265) (VAR(L),L=KK,KKK)
  FORMAT(15X,10E12.5)
270 CONTINUE

```

30860
30870
30880
30890
30900
30910
30920
30930
30940
30950
30960
30970
30980
30990
31000
31010
31020
31030
31040
31050
31060
31070
31080
31090
31100
31110
31120
31130
31140
31150
31160
31170
31180
31190
31200
31210
31220
31230
31240
31250
31260
31270
31280
31290
31300
31310
31320
31330


```

31340 WRITE(IOUT2,3)
31350 FORMAT(/,1X,RADIUS(J),)
31360 WRITE(IOUT2,265) (RR(K),K=1,NDR)
31370 IF(NDIM.EQ.1) GO TO 5
31380 WRITE(IOUT2,4)
31390 FORMAT(/,1X,AXIAL DISTANCE(K),)
31400 KK=NDR+1
31410 KKK=NDR+NDZ
31420 WRITE(IOUT2,265) (RZ(K),K=KK,KKK)
31430 CONTINUE
31440 RETURN
31450 END
31460

100 SUBROUTINE QVRISU(ISUB,ISU,JJ,KK)
COMMON/C3/ DA(175),DADI(25),VAR(3500)
COMMON/C6/ ISTEP,IV(10),IVV(10),IAV(10)
EQUIVALENCE (DA(1),NDIM),(DA(17),IFROM), (DA(67),INTCON)
GO TO (100,100,300,300,600),ISUB
CONTINUE
CALL STRUP(ISUB,IPU,JJ,KK)
IF(IPU.NE.0) GO TO 1000
IF(IJJ.EQ.3) GO TO 1000
IF(IIFROM.NE.0) GO TO 110
CALL MESH
IF( INTCON.LT.4) CALL INIT1
IF( INTCON.EQ.4) CALL INIT4
IF( INTCON.EQ.5) CALL INIT5
CONTINUE
110 CALL RUNDAT
CALL RBC
IF(NDIM.EQ.2) CALL ZBC
GO TO 1000
CONTINUE
300 GO TO 1000
600 CONTINUE
IF(IJJ.EQ.0) GO TO 1000
IF(IJJ.EQ.1) CALL OUTPI
IF(IAV(6).EQ.1) CALL IEPLT
1000 CONTINUE
RETURN
END
31470
31480
31490
31500
31510
31520
31530
31540
31550
31560
31570
31580
31590
31600
31610
31620
31630
31640
31650
31660
31670
31680
31690
31700
31710
31720
31730
31740

```



```

30 DO 30 M=1,107
PT(M)=COMT6
PT(1)=COMT7
AP=AMAX-(L-1)*DA+0.5*DA
AM=AP-DA
IF((AP-GT.O.).AND.(AM.LE.O.)) GO TO 40
IF((AP.GT.AMAX).AND.(AM.LE.AMAX)) GO TO 60
IF((AP.GT.AMIN).AND.(AM.LE.AMIN)) GO TO 70
GO TO 80
40 Y=0.
IPT=1
DO 50 M=1,107
PT(M)=COMT8
GO TO 80
60 Y=AMAX
IPT=1
GO TO 80
70 Y=AMIN
IPT=1
80 CONTINUE
MM=1
DO 100 N=1,5
IF(INP(N).EQ.O.) GO TO 100
IF(MM=MM+NP(N)-1
MM=MM
DO 90 M=MM,MMM
IF((A(M).GE.AP).OR.(A(M).LT.AM)) GO TO 90
IR=INT((R(M)-RMIN)/DR+0.5)+1
GO TO (81,82,83,81,81), N
81 PT(IR)=PC(N)
GO TO 85
82 IF(PT(IR).NE.PC(1)) PT(IR)=PC(2)
IF(PT(IR).EQ.PC(1)) PT(IR)=COMT9
GO TO 85
83 IF(PT(IR).EQ.COMT6) PT(IR)=PC(3)
IF(PT(IR).EQ.COMT8) PT(IR)=PC(3)
IF(PT(IR).EQ.PC(1)) PT(IR)=COMT10
IF(PT(IR).EQ.PC(2)) PT(IR)=COMT11
IF(PT(IR).EQ.COMT6) PT(IR)=COMT12
GO TO 85
85 CONTINUE
90 CONTINUE
MM=MM+1
100 CONTINUE
IF(PT.EQ.1) GO TO 120
WRITE(IOUT2,110) (PT(M),M=1,107)
110 FORMAT(24X,107A1)
GO TO 140
120 WRITE(IOUT2,130) Y,(PT(M),M=1,107)

```



```

130 FORMAT(10X,E13.6,'-',107A1)
140 CONTINUE
150 WRITE(IOUT2,150) RMIN,RMAX
150 FORMAT(24X,'I',E11.3,88X,E10.3,'I')
150 RETURN
150 END

```

```

SUBROUTINE RBC
THIS SUBROUTINE USES FILE MHDOUT.
IMPLICIT REAL*8 (C)

```

C

```

1 COMMON/C3/ DA(175),DADT(25),VAR(3500)
COMMON/C4/ IBCR(10),IBCRZ(10),IBSZ(10),ISHK(10),NZP(3),
1 IMPSHK,JSTEP
COMMON/C6/ISTEP,IV(10),IVV(10),IAV(10)
COMMON/C21/RZ(150)
COMMON/C30/ INP,IOUT1,IOUT2,INP2,IOUT4
1 DIMENSION P(50),RR(10)
EQUIVALENCE (DA(25),NDIM),(DA(9),NDR),(DA(12),DR),(DA(16),NV),
1 EQUIVALENCE (RR,RZ)
DATA
1 COM1 1//DRO/DR=0//,COM1 2//RO=MIN+(//,COM1 3//RO-MIN)*//,
2 COM1 4//EXP(-DT//,COM1 5//TRO//,COM1 6//EXP(VR)*//,
3 COM1 7//DT/RO*//,COM1 8//DR//,COM1 9//EXP(-VR)*//,
4 COM1 10//DRO/DT=0//,COM1 11//ID CONT//,COM1 12//EQ//,
5 COM1 13//VR=0//,COM1 14//VR=5*VR//,COM1 15//(+//,
6 COM1 16//DVR/DR=0//,COM1 17//OHM S.LA//,COM1 18//W,Z COMP//,
7 COM1 19//,E=0//,COM1 20//DVR/DT=0//,COM1 21//(-//,
8 COM1 22//VP=0//,COM1 23//DVP/DR=0//,COM1 24//VP=5*VP//,
9 COM1 25//DVZ/DR=0//,COM1 26//VZ=5*VZ//,COM1 27//VZ=0//,
A COM1 28//DT/DR=0//,COM1 29//DT/DT=0//,COM1 30//DI/DR=0//,
B COM1 31//DPI/DT=0//,COM1 32//DI/DR=0//,COM1 33//DI/DT=0//,
COM1 34//DPE/DR=0//,COM1 35//DPE/DT=0//,COM1 36//BR=0//,
COM1 37//DIV/DT=0//,COM1 38//BP=0//,COM1 39//BP=INIT+//,
COM1 40//MAX-INI//,COM1 41//I)*SIN(2//,COM1 42//P#I//,
1 COM1 43//I2=0//,COM1 44//DBZ/DR=0//,COM1 45//W,PHI CO//,
2 COM1 46//MP, EP=//,COM1 47//,COM1 48//DBZ/DT=0//,
3 COM1 49//BZ=INIT+//,COM1 50//(-INIT+//,COM1 51//5*MAX*(R//,
4 COM1 52//M+1-(RM-//,COM1 53//1)*SIN(P//,COM1 54//1*(Z/ZMA//,
5 COM1 55//X-5//)/R//,COM1 56//M)*SIN(2//,COM1 57//MAX*SIN//,
6 COM1 58//2*PI*F#I//,COM1 59//,COM1 60//BZ=MAX*S//,
7 COM1 61//IN(2*PI*//,COM1 62//F#I+DEL//,COM1 63//DSI/DR=0//,
8 COM1 64//AP=0//,COM1 65//DSB/DR=0//,COM1 66//SI=0//,
9 COM1 67//
DO 2030 K=1,2
M=0
DO 2010 J=1,NV

```



```

JJ=IVV(JJ)
IBCRJ=IBCR(JJ)
IBCRJJ=IBCR(JJ)
IAVJ=IAV(JJ)
IF(K.EQ.1) GO TO (100,200,300,400,500,600,700,800,900,1000),JJ
GO TO (150,250,350,450,550,650,750,850,950,1050),JJ
100 GO TO (101,102,103),IBCRJ
101 M=M+1
GO TO 2000
102 P(M+1)=COMT2
P(M+2)=COMT3
M=M+2
IF(K.EQ.1) GO TO (110,110,103),IBCRJ
110 P(M+1)=COMT4
P(M+2)=COMT5
M=M+2
GO TO 2000
103 P(M+1)=COMT6
P(M+2)=COMT7
P(M+3)=COMT8
IF(K.EQ.1) P(M+1)=COMT9
M=M+3
GO TO 2000
104 P(M+1)=COMT10
M=M+1
GO TO 2000
150 GO TO (102,102,101,154,104),IBCRJ
154 P(M+1)=COMT11
P(M+2)=COMT12
M=M+2
GO TO 2000
200 GO TO (201,202,201,204,205,206),IBCRJJ
201 P(M+1)=COMT13
M=M+1
GO TO 2000
202 P(M+1)=COMT14
P(M+2)=COMT15
M=M+2
GO TO 2000
204 P(M+1)=COMT16
M=M+1
GO TO 2000
205 P(M+1)=COMT17
P(M+2)=COMT18
P(M+3)=COMT19
M=M+3

```

33150
33160
33170
33180
33190
33200
33210
33220
33230
33240
33250
33260
33270
33280
33290
33300
33310
33320
33330
33340
33350
33360
33370
33380
33390
33400
33410
33420
33430
33440
33450
33460
33470
33480
33490
33500
33510
33520
33530
33540
33550
33560
33570
33580
33590
33600
33610
33620


```

206      GO TO 2000
          P(M+1)=COMT20
          M=M+1
          GO TO 2000
250      GO TO (251,204,201,154,206),IBCRJ
251      P(M+1)=COMT14
          P(M+2)=COMT21
          M=M+2
          GO TO 2000
300      CONTINUE
301      P(M+1)=COMT22
          M=M+1
          GO TO 2000
350      GO TO (351,352,301),IBCRJ
351      P(M+1)=COMT23
          M=M+1
          GO TO 2000
352      P(M+1)=COMT24
          P(M+2)=COMT21
          M=M+2
          GO TO 2000
400      CONTINUE
401      P(M+1)=COMT25
          M=M+1
          GO TO 2000
450      GO TO (401,452,453),IBCRJ
452      P(M+1)=COMT26
          P(M+2)=COMT21
          M=M+2
          GO TO 2000
453      P(M+1)=COMT27
          M=M+1
          GO TO 2000
500      IF (IAV(5).EQ.1) GO TO 510
          GO TO (501,502),IBCRJ
501      P(M+1)=COMT28
          M=M+1
          GO TO 2000
502      P(M+1)=COMT29
          M=M+1
          GO TO 2000
510      GO TO (511,512),IBCRJ
511      P(M+1)=COMT30
          M=M+1
          GO TO 2000
512      P(M+1)=COMT31
          M=M+1
          GO TO 2000

```

```

33630
33640
33650
33660
33670
33680
33690
33700
33710
33720
33730
33740
33750
33760
33770
33780
33790
33800
33810
33820
33830
33840
33850
33860
33870
33880
33890
33900
33910
33920
33930
33940
33950
33960
33970
33980
33990
34000
34010
34020
34030
34040
34050
34060
34070
34080
34090
34100

```


550	IF (IAVJ-EQ.1) GO TO 560	34110
560	GO TO (502,501), IBCRJ	34120
560	IF IAV(6).EQ.1, IBCRJ	34130
600	GO TO (601,602), IBCRRJ	34140
601	P(M+1)=COMT32	34150
	M=M+1	34160
	GO TO 2000	34170
602	P(M+1)=COMT33	34180
	M=M+1	34190
	GO TO 2000	34200
610	GO TO (611,612), IBCRRJ	34210
611	P(M+1)=COMT34	34220
	M=M+1	34230
	GO TO 2000	34240
612	P(M+1)=COMT35	34250
	M=M+1	34260
	GO TO 2000	34270
650	IF (IAVJ-EQ.1) GO TO 660	34280
660	GO TO (602,601), IBCRJ	34290
700	CONTINUE	34300
701	P(M+1)=COMT36	34310
	M=M+1	34320
	GO TO 2000	34330
750	GO TO (751,701), IBCRJ	34340
751	P(M+1)=COMT37	34350
	M=M+1	34360
	GO TO 2000	34370
800	GO TO (801,802,803,801), IBCRRJ	34380
801	P(M+1)=COMT38	34390
	M=M+1	34400
	GO TO 2000	34410
802	P(M+1)=COMT39	34420
	P(M+2)=COMT40	34430
	P(M+3)=COMT41	34440
	P(M+4)=COMT42	34450
	M=M+4	34460
	GO TO 2000	34470
803	P(M+1)=COMT43	34480
	M=M+1	34490
	GO TO 2000	34500
850	GO TO (802,803,801,205), IBCRJ	34510
900	GO TO (901,902,903), IBCRRJ	34520
901	P(M+1)=COMT44	34530
	M=M+1	34540
	GO TO 2000	34550
902	P(M+1)=COMT17	34560
		34570
		34580

P(M+2)=COMI45	34590
P(M+3)=COMI46	34600
M=M+3	34610
IF(K.EQ.2) GO TO (952,910), IBCRJ	34620
910 M=M+1	34630
GO TO 2000	34640
P(M+1)=COMI48	34650
M=M+1	34660
GO TO 2000	34670
M=M+1	34680
GO TO 2000	34690
950 GO TO (951,902,902,954,903), IBCRJ	34700
951 IF(NDIM.EQ.2) GO TO 961	34710
P(M+1)=COMI49	34720
P(M+2)=COMI40	34730
P(M+3)=COMI41	34740
P(M+4)=COMI42	34750
M=M+4	34760
GO TO 2000	34770
961 P(M+1)=COMI49	34780
P(M+2)=COMI50	34790
P(M+3)=COMI51	34800
P(M+4)=COMI52	34810
P(M+5)=COMI53	34820
P(M+6)=COMI54	34830
P(M+7)=COMI55	34840
P(M+8)=COMI56	34850
P(M+9)=COMI42	34860
M=M+9	34870
GO TO 2000	34880
952 P(M+1)=COMI57	34890
P(M+2)=COMI58	34900
P(M+3)=COMI59	34910
M=M+3	34920
GO TO 2000	34930
954 P(M+1)=COMI60	34940
P(M+2)=COMI61	34950
P(M+3)=COMI62	34960
M=M+3	34970
GO TO 2000	34980
1000 GO TO (1001,1002,1002), IBCRRJ	34990
1001 CONTINUE	35000
IAPV1=IAPV(10)+1	35010
GO TO (1011,1012,1013), IAPV1	35020
1011 P(M+1)=COMI63	35030
M=M+1	35040
GO TO 2000	35050
1012 P(M+1)=COMI64	35060
M=M+1	


```

1013 GO TO 2000
      P(M+1)=COMT65
      GO TO 2000
1002 P(M+1)=COMT66
      M=M+1
1050 GO TO 2000
2000 P(M+1)=COMT67
      GO TO 2000
2010 CONTINUE
      IF(K.EQ.1) WRITE(IOUT2,2020) RMIN,(P(J),J=1,M)
2020 FORMAT(5X,'BOUNDARY CONDITIONS AT R=,E12.5,---,4X,10A8/,3(8X,15
      1A8,/))
      RMAX=RR(NDR)
      IF(K.EQ.2) WRITE(IOUT2,2020) RMAX,(P(J),J=1,M)
2030 CONTINUE
      RETURN
      END

```

```

C
SUBROUTINE RUNDAT
THIS SUBROUTINE USES FILE MHDOUT.
REAL*8 IDA
COMMON/C3/ DA(175),DADT(25),VAR(3500)
COMMON/C30/ INP,IOUT1,IOUT2,INP2,IOUT4
DIMENSION IDA(40),ADDA(40),JBCRR(2),JBCR(2),JBCZZ(2),JBCZ(2),
1 I1SHOCK(2)
DIMENSION RR(10)
EQUIVALENCE
  (DA(1),NDIME),(DA(2),IVERS),(DA(3),MAT),(DA(4),ITC),
  (DA(5),IDATE),(DA(6),JBCR),(DA(8),NDT),(DA(12),DRMIN),
  (DA(9),NDR),(DA(10),NDZ),(DA(11),DT),(DA(12),DRMIN),
  (DA(13),DZMIN),(DA(14),IPIVOT),(DA(15),ISTEPO),
  (DA(17),IFROM),(DA(18),STRYM),(DA(19),IFYL),
  (DA(20),NADDA),(DA(21),IV),(DA(23),ITAPE),
  (DA(24),I8E),(DA(25),RMIN),(DA(26),ADDA(1)),
  (DA(52),MATALI),(DA(53),I1AV),(DA(55),IDVIX),
  (DA(56),NREG),(DA(57),Z2),(DA(58),NDZ2),(DA(59),R2),
  (DA(60),NDR2),(DA(61),Z1),(DA(62),NDZ1),(DA(63),R1),
  (DA(64),NDR1),(DA(65),ZMAX),(DA(66),RMAX),
  (DA(67),INTCON),(DA(68),JBCZZ),
  (DA(71),IDA(1)),(DA(156),JDDDR),
  (DA(158),NTCDRV),(DA(164),ICCOORD),
  (DA(166),JBCRR),(DA(168),JBCZ),(DA(165),ISOI),
  (DA(172),I1SHOCK),
  (DA(170),JDDZ),
  EQUIVALENCE (RR,RZ)
  DR=RR(2)-RR(1)

```



```

DZ=RZ(NDR+2)-RZ(NDR+1)
IF(IFROM.EQ.0) WRITE(IOUT2 ,130) ITAPE,IFYL
130 FORMAT(1,FILE NUMBER,15,-,12,/)
IF(IFROM.NE.0) WRITE(IOUT2 ,135) ITAPE,IFYL,IFROM,STR1YM
135 FORMAT(1,FILE NUMBER,15,-,12,10X,'CONTINUATION OF FILE',15,' AT
1 TIME=',E15.8,/)
WRITE(IOUT2 ,140) NDIM,IVERS,MAT,INITCON,ITC,IIV,IIAV,IDATE,
1 IDVDX,MAT,ITISHOCK
140 1 FORMAT(5X,MHD,11,D-,413, (IIV),110, (IIAV),110, DATE ,
15, (IDVDX),15, (MALALI),15, SHOCKS=,215)
145 WRITE(IOUT2 ,145) NDT,DT,NDR,DRMIN,JBCRR,JBCR,JJDDR
145 FORMAT(5X,NDT=,14,2X,DT=,E10.3,2X,NDR=,13,2X,DR=,E10.3,2X,
1 INNER RADIAL BC=,215,2X,OUTER RADIAL BC=,215,2X,R DERIVS=,11
1)
IF(NDIM.EQ.2) WRITE(IOUT2 ,147) NDZ,DZMIN,JBCZZ,JBCZ,JJDDZ
147 FORMAT(3X,NDZ=,13,2X,DZ=,E10.3,2X,LOWER AXIAL BC=,215,2X,
1 UPPER AXIAL BC=,215,2X,Z DERIVS=,110)
IF(NREG.EQ.1) GO TO 230
IF(NREG.EQ.3) GO TO 220
DZ=RR(NDR)-RR(NDR-1)
NDRR=NDR-NDK1
WRITE(IOUT2 ,200) RMIN,NDR1,DR,R1,NDRR,DR2,RMAX
200 FORMAT(1X,NONUNIFORM R MESH=,E10.3,3(5X,15,,' ,E10.3,5X,E10.3))
IF(NDIM.EQ.1) GO TO 230
DZ2=RZ(NDR+NDZ)-RZ(NDR+NDZ-1)
NDZ2=NDZ-NDZ1
WRITE(IOUT2 ,210) NDZ1,DZ,Z1,NDZ2,DZ2,ZMAX
210 1 FORMAT(1X,NONUNIFORM Z MESH= 0.000+000',3(5X,15,,' ,E10.3,5X,E10.3)
1)
GO TO 230
DZ1=RR(NDR+1)-RR(NDR1)
DZ2=RR(NDR)-RR(NDR-1)
DZ=NDR-NDR2
WRITE(IOUT2 ,200) RMIN,NDR1,DR,R1,NDR2,DR1,R2,NDRR,DR2,RMAX
IF(NDIM.EQ.1) GO TO 230
DZ1=RZ(NDR+NDZ1+1)-RZ(NDR+NDZ1)
DZ2=RZ(NDR+NDZ)-RZ(NDR+NDZ-1)
NDZ2=NDZ-NDZ2
WRITE(IOUT2 ,210) NDZ1,DZ,Z1,NDZ2,DZ1,Z2,NDZ2,DZ2,ZMAX
230 CONTINUE
WRITE(IOUT2 ,150) (IDA(J),ADDA(J),J=1,NADDA)
150 FORMAT(15(5X,A8,E10.3))
IF(PIVOT.NE.0) WRITE(IOUT2 ,160)
160 FORMAT(5X,PIVOT PIVOTING USED IN GAUSS ELIMINATION')
IF((ISTEP0.NE.0).AND.(IFROM.EQ.0)) WRITE(IOUT2 ,170)
170 FORMAT(5X,IMPLICIT IN Z DIRECTION FIRST)
IF(IDVDX.EQ.1) WRITE(IOUT2 ,180)

```



```

180 FORMAT(5X,'FOURTH ORDER DERIVATIVES USED')
185 IF(IBE.EQ.1) WRITE(IOUT2,195)
195 FORMAT(5X,'MAGNETIC FIELD SEPARATED INTO PLASMA AND EXTERNAL COMPO
196 NENTS')
196 IF(ISO.EQ.1) WRITE(IOUT2,196)
196 FORMAT(5X,'ANISOTROPIC TRANSPORT COEFFICIENTS USED')
197 IF(TCOORD.EQ.1) WRITE(IOUT2,197)
197 FORMAT(5X,'EQUATIONS SOLVED IN SPHERICAL COORDINATES')
198 IF(TCOORD.EQ.-1) WRITE(IOUT2,198)
198 FORMAT(5X,'EQUATIONS SOLVED IN RECTANGULAR COORDINATES')
199 IF(NTCDRV.EQ.1) WRITE(IOUT2,300)
300 FORMAT(5X,'TRANSPORT COEFFICIENT DERIVATIVES SET TO ZERO')
RETURN
END

SUBROUTINE SETUP(J,K)
COMMON/C3/ DAI(15),DAPT(25),VAR(3500)
COMMON/C6/ISTEP,IV(10),IVV(10),IAV(10)
COMMON/C8/ICDM(4),TCRM(4),IC(4),TCRP(4),TCZP(4),DICOR(4),DICDZ(4),
1 DICDVM(4,10),DICDV(4,10),DICDVP(4,10),DIDVDX(4,10),W(4),
2 W(4)
1 COMMON/C9/VARIZM(10),VARIRM(10),VARI(10),VARIRP(10),VARIZP(10),
1 DVDR(10),DVDZ(10),D2VDR2(10),D2VDZ2(10)
1 COMMON/C10/RM,R,RP,DRM,DRP,DRMP,DRPM,DDRPM,
1 ZM,Z,RP,DZP,DZPM,DDZPM
1 COMMON/C11/BERM(4),BERM(4),BE(4),BERP(4),BERZM(4),
1 BERPM(4),BERP(4),BERP(4),BERZP(4),DBEDR(4),DBEDZ(4),
2 DBEDR(4),DBEDZ(4)
1 COMMON/C13/IDDX(10)
1 COMMON/C15/ICDZM(4),TCORM(4),TCD(4),TCDRP(4),TCDZP(4)
1 COMMON/C21/RZ(150)
1 COMMON/C22/JDDR(10),JDDZ(10)
1 COMMON/C25/CVRMZP(10),CVRMZM(10),CVRPZP(10),CVRPZM(10)
1 COMMON/C26/BERMZP(4),BERMZP(4),BERMZM(4),BERPZM(4)
1 COMMON/C27/TCRMZP(4),TCRPZP(4),TCRMZM(4),TCRPZM(4),
1 TDRMZP(4),TDRPZP(4),TDRMZM(4),TDRPZM(4)
1 DIMENSION DUM(4,10)
1 DIMENSION RR(10)
1 DIMENSION VVAR(10,5),DV(10,4)
1 DIMENSION X(5),CBX(5,2),CBR(5),CBZ(5),CCX(5),CCZ(5)
1 EQUIVALENCE (CBX(1),CBR(1)),(CBX(6),CBZ(1)),(CCX(1),CCZ(1)),
1 (CCX(6),CCZ(1))
1 EQUIVALENCE (DA(1),NDM), (DA(4),NTC), (DA(9),NDR), (DA(10),NDZ),
1 (DA(16),NV), (DA(24),IBE), (DA(31),BZINT),
1 (DA(54),IPLAVC), (DA(165),ISO)
3 EQUIVALENCE (DUM,OLDRZ)

```



```

EQUIVALENCE (RZ,RR)
EQUIVALENCE (VVARI(1),VARIZM(1)),(DV(1),DVDR(1))
RM=RR(J-1)
R=RR(J)
RP=RR(J+1)
DRP=RP-R
DRM=R-RM
DDRPM=(DRP-DRM)/(DRM*DRP)
DRPM=DKM/(DRP*(DRM+DRP))
DRPM=DRP/(DRM*(DRM+DRP))
IF(ABS(1.-DRP/DRM).GT.1.0E-6) GO TO 5
DRP=DRM
DDRPM=0
DRPM=0.5/DRP
DRMP=DRPM
CONTINUE
5 IF(NDIM.EQ.1) GO TO 10
KZ=NDIM+K
ZM=RZ(KZ-1)
Z=RZ(KZ)
ZP=RZ(KZ+1)
DZP=ZP-Z
DZM=Z-ZM
DDZPM=(DZP-DZM)/(DZM*DZP)
DZMP=DZM/(DZP*(DZM+DZP))
DZPM=DZP/(DZM*(DZM+DZP))
IF(ABS(1.-DZP/DZM).GT.1.0E-6) GO TO 10
DZP=DZM
DDZPM=0
DZPM=0.5/DZP
DZMP=DZPM
CONTINUE
10 IDDR=0
IDDZ=0
IF(1DVDX.NE.1) GO TO 50
DO 45 N=1,NDIM
IF(N.EQ.2) GO TO 315
IF((J-GE.(NDR-1)).OR.(J.LE.2)) GO TO 45
IDDR=1
X(2)=RR(J-2)-R
X(3)=RR(J-1)-R
X(4)=RR(J+1)-R
X(5)=RR(J+2)-R
GO TO 310
315 IF((K-GE.(NDZ-1)).OR.(K.LE.2)) GO TO 45
IDDZ=1
KZ=NDR+K
Z=RZ(KZ)

```

36460
36470
36480
36490
36500
36510
36520
36530
36540
36550
36560
36570
36580
36590
36600
36610
36620
36630
36640
36650
36660
36670
36680
36690
36700
36710
36720
36730
36740
36750
36760
36770
36780
36790
36800
36810
36820
36830
36840
36850
36860
36870
36880
36890
36900
36910
36920
36930


```

36940 X(2)=RZ(KZ-2)-Z
36950 X(3)=RZ(KZ-1)-Z
36960 X(4)=RZ(KZ+1)-Z
36970 X(5)=RZ(KZ+2)-Z
36980 CONTINUE
36990 CBX(1,N)=0.
37000 CCX(1,N)=0.
37010 CCX(2,N)=X(3)*X(4)+X(3)*X(5)+X(4)*X(5)
37020 CCX(3,N)=X(2)*X(4)+X(2)*X(5)+X(4)*X(5)
37030 CCX(4,N)=X(2)*X(3)+X(2)*X(5)+X(3)*X(5)
37040 CCX(5,N)=X(2)*X(3)+X(2)*X(4)+X(3)*X(4)
37050 DO 25 L=2,5
37060 CBX(1,N)=CBX(1,N)-1./X(L)
37070 CBX(1,N)=1./X(L)
37080 CCX(1,N)=2.*CCX(L,N)/X(L)
37090 DO 15 M=2,5
37100 IF(L.EQ.M) GO TO 15
37110 CCX(L,N)=CCX(L,N)/(X(L)-X(M))
37120 CBX(L,N)=CBX(L,N)*X(M)/(X(L)-X(M))
37130 LL=L+1
37140 CONTINUE
37150 M=LL,5
37160 DO 320 CCX(1,N)=CCX(1,N)+2./(X(L)*X(M))
37170 CONTINUE
37180 DO 45 CONTINUE
37190 JK=J+NDR*(K-1)
37200 INC=NDR*NDZ
37210 DO 30 L=1,NV
37220 LL=1+V(L)
37230 VAR(1,LL)=VAR(JK)
37240 VARIRP(LL)=VAR(JK+1)
37250 VARIRM(LL)=VAR(JK-1)
37260 IF(1DDR.EQ.0) GO TO 111
37270 DVDR(LL)=CBR(1)*VAR(JK)+CBR(2)*VAR(JK-1)+CBR(
14)*VAR(JK+1)+CBR(5)*VAR(JK+2)
37280 D2VDR2(LL)=CBR(1)*VAR(JK)+CBR(2)*VAR(JK-1)+CC
1R(4)*VAR(JK+1)+CCR(5)*VAR(JK+2)
37290 GO TO 112
37300 CONTINUE
37310 DOVDR(LL)=DRMP*VARIRP(LL)-DRPM*VARIRM(LL)+DDRPM*VARI(LL)
37320 D2VDR2(LL)=2.*(VARIRP(LL)/DRP+VARIRM(LL)/DRM)-2.*VARI(LL
111 CONTINUE
37330 D2VDR2(LL)=2.*(VARIRP(LL)/DRP+VARIRM(LL)/DRM)-2.*VARI(LL
112 CONTINUE
37340 IF(NDIM.EQ.1) GO TO 20
37350 VARIZP(LL)=VAR(JK+ND5)
37360 VARIZM(LL)=VAR(JK-ND5)
37370 IF(1DDZ.EQ.0) GO TO 116
37380
37390
37400
37410

```



```

DVBZ(LL)=CBZ(1)*VAR(JK)+CBZ(2)*VAR(JK-2*NDR)+CBZ(3)*VAR(JK-NDR
1)*CBZ(4)*VAR(JK+NDR)+CBZ(5)*VAR(JK+2*NDR)
DZVDZ2(LL)=CGZ(1)*VAR(JK)+CGZ(2)*VAR(JK-2*NDR)+CGZ(3)*VAR(JK-N
1DR)+CGZ(4)*VAR(JK+NDR)+CGZ(5)*VAR(JK+2*NDR)
GO TO 117
116 CONTINUE
DVBZ(LL)=DZMP*VARIZP(LL)-DZPM*VARIZM(LL)+DZP*VARIZM(LL)-2.*VARI(LL
1)/(DZP*DZM)
DZVDZ2(LL)=2.*(VARIZP(LL)/DZP+VARIZM(LL)/DZM)/(DZP+DZM)-2.*VARI(LL
1)/(DZP*DZM)
117 CONTINUE
GO TO 30
20 VARIZP(LL)=VARI(LL)
VARIZM(LL)=VARI(LL)
DVBZ(LL)=0.
DZVDZ2(LL)=0.
30 JK=JK+INC
NVP=NVP+1
DO 40 L=NVP,10
LL=1VV(L)
DO 31 M=1,5
VARI(LL,M)=0.
31 VARI(LL,M)=0.
DO 32 M=1,4
DV(LL,M)=0.
32 DV(LL,M)=0.
40 CONTINUE
DO 405 L=1,NTC
KH(L)=0.
DO 410 L=1,10
DO 407 M=1,NTC
407 DTDX(M,L)=0.
410 IDDX(L)=0
L=1VV(L)
DO 420 LL=1,NV
L=1VV(LL)
IF(JDDR(L).NE.0) CALL ONESID(1,L)
IF(JDDZ(L).NE.0) CALL ONESID(2,L)
420 CONTINUE
IF(NDIM.EQ.1) GO TO 43
JK=J+NDR*(K-1)
INC=NDR*NDZ
DO 41 L=1,NV
LL=1VV(L)
CVRMZP(LL)=VAR(JK+NDR-1)
CVRMZP(LL)=VAR(JK+NDR+1)
CVRMZM(LL)=VAR(JK-NDR-1)
CVRMZM(LL)=VAR(JK-NDR+1)
41 JK=JK+INC
DO 42 L=NVP,10
LL=1VV(L)
CVRMZP(LL)=0.

```

37420
37430
37440
37450
37460
37470
37480
37490
37500
37510
37520
37530
37540
37550
37560
37570
37580
37590
37600
37610
37620
37630
37640
37650
37660
37670
37680
37690
37700
37710
37720
37730
37740
37750
37760
37770
37780
37790
37800
37810
37820
37830
37840
37850
37860
37870
37880
37890


```

CVRPZP(LL)=0.
CVRMZM(LL)=0.
CVRPZM(LL)=0.
42 CONTINUE
43 IF(ISO.NE.1) GO TO 55
  IF(IV(10).GT.NV) GO TO 55
  CALL BFRMSI(J,K,VARI(7),VARI(9))
  JP=J+1
  JM=J-1
  KP=K+1
  KM=K-1
  CALL BFRMSI(JM,K,VARIRM(7),VARIRM(9))
  CALL BFRMSI(JP,K,VARI(7),VARI(9))
  IF(NDIM.EQ.1) GO TO 55
  CALL BFRMSI(JM,KP,CVRMZP(7),CVRMZP(9))
  CALL BFRMSI(JP,KP,CVRPZP(7),CVRPZP(9))
  CALL BFRMSI(JM,KM,CVRPZM(7),CVRPZM(9))
  CALL BFRMSI(JM,KM,CVRMZM(7),CVRMZM(9))
  CALL BFRMSI(J,KP,VARI(7),VARI(9))
  CALL BFRMSI(J,KM,VARI(7),VARI(9))
  CONTINUE
55 CALL BEXT(J,K)
  GO TO (51,52,53,54),ISTEP
51 IF(J.NE.2) GO TO 60
  CALL TRANCO(VARI,TC,DTCDV,0,BERM,TCDRM)
  CALL TRANCO(VARI,TC,DTCDV,0,BE,ICD)
  IF(NDIM.EQ.1) GO TO 90
  CALL TRANCO(VARI,ZP,ICZP,DUM,1,BE,ZP,ICDZP)
  CALL TRANCO(VARI,ZM,ICZM,DUM,1,BE,ZM,ICDZM)
  CALL TRANCO(CVRMZP,ICRMZM,DUM,1,BERMZM,TCRMZM)
  CALL TRANCO(CVRMZM,ICRMZM,DUM,1,BERMZM,TCRMZM)
  GO TO 90
52 IF(K.NE.2) GO TO 60
  CALL TRANCO(VARI,ZM,TCZM,DTCDVM,0,BE,ZM,TCDZM)
  CALL TRANCO(VARI,TC,DTCDV,0,BE,ICD)
  CALL TRANCO(VARI,ZP,ICRP,DUM,1,BERP,TCDRP)
  CALL TRANCO(VARI,ZM,ICRM,DUM,1,BERM,TCDRM)
  CALL TRANCO(CVRMZM,ICRMZM,DUM,1,BERMZM,TCRMZM)
  CALL TRANCO(CVRPZM,ICRPZM,DUM,1,BERPZM,TCRPZM)
  GO TO 120
53 IF(J.NE.1) GO TO 130
  CALL TRANCO(VARI,ZP,ICZP,DUM,1,BE,ZP,ICDZP)
  CALL TRANCO(VARI,TC,DTCDV,0,BE,ICD)
  IF(NDIM.EQ.1) GO TO 160
  CALL TRANCO(VARI,ZP,ICZP,DUM,1,BE,ZP,ICDZP)
  CALL TRANCO(VARI,ZM,ICZM,DUM,1,BE,ZM,ICDZM)
  CALL TRANCO(CVRPZM,ICRPZM,DUM,1,BERPZM,TCRPZM)
  CALL TRANCO(CVRPZP,ICRPZP,DUM,1,BERPZP,TCRPZP)

```



```

GO TO 160
54 IF K-NE.(NDZ-1)) GO TO 130
   CALL TRANCO(VARI ZP,TCZP,DTCDVP,0,BE ZP,TC DZP)
   CALL TRANCO(VARI,TC,DTCDV,0,BE,TC D)
   CALL TRANCO(VARI RP,TCRP,DUM,1,BE RP,TC DRP)
   CALL TRANCO(VARI RM,TCRM,DUM,1,BE RM,TC DRM)
   CALL TRANCO(CVRMZP,TCRMZP,DUM,1,BE RMZP,TC DRMZP)
   CALL TRANCO(CVRPZP,TCRPZP,DUM,1,BE RPZP,TC DRPZP)
   GO TO 210
60 IF(NTCDRV.EQ.1) GO TO 71
   DO 70 L=1,NTC
   DTCDVM(L,M)=DTCDV(L,M)
   DTCDVP(L,M)=TC DRPZP(L,M)
70 CONTINUE
71 IF(1 STEP.EQ.2) GO TO 100
   DO 80 L=1,NTC
   TC DRM(L)=TC D(L)
   TC RM(L)=TC R(L)
   TC(L)=TC RP(L)
80 IF(INDIM.EQ.1) GO TO 90
   DO 92 L=1,NTC
   TC RMZP(L)=TC ZP(L)
   TC ZP(L)=TC RPZP(L)
   TC DRMZP(L)=TC DZP(L)
   TC DZP(L)=TC RPZP(L)
   TC RMZM(L)=TC ZM(L)
   TC ZM(L)=TC RPZM(L)
   TC DRMZM(L)=TC DZM(L)
   TC DZM(L)=TC RPZM(L)
92 CONTINUE
90 CALL TRANCO(VARI RP,TCRP,DTCDVP,0,BE RP,TC DRP)
   IF(NDIM.EQ.1) GO TO 170
   CALL TRANCO(CVRPZP,TCRPZP,DUM,1,BE RPZP,TC DRPZP)
   CALL TRANCO(CVRPZM,TCRPZM,DUM,1,BE RPZM,TC DRPZM)
   GO TO 170
100 DO 110 L=1,NTC
   TC RMZM(L)=TC ZM(L)
   TC ZM(L)=TC RPZP(L)
   TC RPZM(L)=TC RP(L)
   TC RP(L)=TC RPZP(L)
   TC DRMZM(L)=TC DRM(L)
   TC DRM(L)=TC DRMZP(L)
   TC DRPZM(L)=TC DRP(L)
   TC DRP(L)=TC DRPZP(L)
   TC DZM(L)=TC D(L)
   TC D(L)=TC DZP(L)

```



```

110 TCZM(L)=TC(L)
111 TC(L)=TCZP(L)
112 CALL TRANCO(VARIZP,TCZP,DTCDVP,0,BECP,TCZP)
113 CALL TRANCO(CVRMZP,TCRMZP,DUM,1,BERMZP,TCRMZP)
114 CALL TRANCO(CVRPZP,TCRPZP,DUM,1,BERPZP,TCRPZP)
115 GO TO 220
130 IF (NTCDRV.EQ.1) GO TO 141
131 DO 140 L=1,NTC
132 DO 140 M=1,10
133 DTCDVP(L,M)=DTCDV(L,M)
134 DTCDV(L,M)=DTCDVM(L,M)
135 CONTINUE
141 IF (ISTEP.EQ.4) GO TO 190
142 DO 150 L=1,NTC
143 TCDRP(L)=TCD(L)
144 TCD(L)=TCDRM(L)
145 TCRP(L)=TCR(L)
146 TC(L)=TCRM(L)
147 IF(NDIM.EQ.1) GO TO 160
148 DO 162 L=1,NTC
149 TCRPZP(L)=TCZP(L)
150 TCZP(L)=TCRMZP(L)
151 TDRPZP(L)=TCDZP(L)
152 TCDZP(L)=TDRMZP(L)
153 TDRPZM(L)=TCDZM(L)
154 TCDZM(L)=TDRMZM(L)
155 TCRPZM(L)=TCZM(L)
156 TCZM(L)=TCRMZM(L)
157 CONTINUE
162 CALL TRANCO(VARIRM,TCRM,DTCDVM,0,BERM,TCDRM)
163 IF(NDIM.EQ.1) GO TO 170
164 CALL TRANCO(CVRMZM,TCRMZM,DUM,1,BERMZM,TCRMZM)
165 CALL TRANCO(CVRMZP,TCRMZP,DUM,1,BERMZP,TCRMZP)
166 CONTINUE
170 CONTINUE
175 IF (NTCDRV.EQ.1) GO TO 240
176 DO 180 L=1,NTC
177 DO 180 MM=1,NV
178 M=IVV(MM)
179 DTDVDX(L,M)=DRMP*DTCDVP(L,M)-DRPM*DTCDVM(L,M)+DDRP*DTCDV(L,M)
180 WW(L)=WW(L)-DTDVDX(L,M)*VARI(M)-DTCDV(L,M)*DVDR(M)
181 DO 200 L=1,NTC
182 TCRMZP(L)=TCRM(L)
183 TCRV(L)=TCRMZM(L)
184 TCRPZP(L)=TCRPZ(L)
185 TCRP(L)=TCRPZM(L)
186 TDRMZP(L)=TCDRM(L)

```



```

TCDRM(L)=IDRMZM(L)
TDRPZP(L)=TCDZP(L)
TCDRP(L)=TDRPZM(L)
TCDZP(L)=TCD(L)
TCDL(L)=TCDZM(L)
TCZP(L)=TC(L)
TC(L)=TCZM(L)
200 CALL TRANCO(VARIZM,TCZM,DTCDVM,O,BEZM,TCZM)
210 CALL TRANCO(CVBMZM,TCRMZM,DUM,I,BERMZM,IDRMZM)
CALL TRANCO(CVRPZM,TCRPZM,DUM,I,BERPZM,IDRPZM)
CONTINUE
220 IF(NTCDRV.EQ.1) GO TO 240
DO 230 L=1,NTC
MM=1,NV
M=1V(M)
DTVDX(L,M)=DZMP*DTCDVP(L,M)-DZPM*DTCDVM(L,M)+DDZPM*DTCDV(L,M)
230 WW(L)=W(L)-DTVDX(L,M)*VARI(M)-DTCDV(L,M)*DVDZ(M)
240 W(L)=TC(L)
TCDRL(L)=DRMP*TCRP(L)-DRPM*TCRM(L)+DDRPM*TC(L)
IF(NDIM.EQ.1) GO TO 245
DTCDZ(L)=DZMP*TCZP(L)-DZPM*TCZM(L)+DDZPM*TC(L)
GO TO 247
245 DTCDZ(L)=0.
247 IF(NTCDRV.EQ.1) GO TO 250
DO 248 M=1,NV
MM=1V(M)
W(L)=W(L)-DTCDV(L,MM)*VARI(MM)
248 CONTINUE
250 CONTINUE
GO TO (255,260,255,260),ISTEP
255 DO 256 L=1,NTC
256 WW(L)=W(L)+DTCDR(L)
GO TO 270
260 DO 261 L=1,NTC
261 WW(L)=WW(L)+DTCDZ(L)
270 CONTINUE
290 RETURN
END

```

39340
39350
39360
39370
39380
39390
39400
39410
39420
39430
39440
39450
39460
39470
39480
39490
39500
39510
39520
39530
39540
39550
39560
39570
39580
39590
39600
39610
39620
39630
39640
39650
39660
39670
39680
39690
39700
39710
39720

39730
39740
39750

SUBROUTINE SHKMAT(J,K)
RETURN
END


```

SUBROUTINE SHOCK(JJJ,KKK)
COMMON/C3/ DAI(175),DADT(25),VAR(3500)
COMMON/C4/ IBCR(10),IBCRZ(10),IB CZ(10),ISHK(10),NZP(3),
1 IMPSHK,JSTEP
COMMON/C6/ ISTEP,IV(10),IVV(10),IAV(10)
COMMON/C7/A(10,10),B(10,10),C(10,10),V(10),COL(10)
COMMON/C21/RZ(150)
DIMENSION OV(9),OVM(9)
EQUIVALENCE (RR,RZ)
1 DIMENSION RR(10)
2 DIMENSION (DA(11),NDIM),(DA(9),NDR),(DA(10),NDZ),(DA(11),DT),
EQUIVALENCE (DA(12),DRMIN),(DA(13),DZMIN),(DA(16),NV),
(DA(25),RMIN),(DA(26),CQA)
EQUIVALENCE (RR,RZ)
DO 100 N=1,NDIM
IF(N.EQ.2) GO TO 20
MM=NDZ-1
IF(NDIM.EQ.1) MM=1
LL=NDZ-1
GO TO 30
20 MM=NDZ-1
LL=NDZ-1
30 CONTINUE
DO 100 M=NDIM,MM
IF(N.EQ.2) GO TO 40
K=M
IST=1+(K-1)*NDR
GO TO 45
40 J=M
JST=J
45 DO 50 JJ=1,NV
50 OV(JJ)=VAR(IST+(JJ-1)*NDR*NDZ)
DO 100 L=2,LL
IF(N.EQ.2) GO TO 60
J=L
DX2=DRMIN*DRMIN
DXP=RR(J+1)-RR(J)
DXM=RR(J)-RR(J-1)
JK=J+(K-1)*NDR
JKP=JK+1
LV=2
GO TO 70
60 K=L
DX2=DZMIN*DZMIN
KZ=NDZ+K
DXP=RZ(KZ+1)-RZ(KZ)
DXM=RZ(KZ)-RZ(KZ-1)
JK=J+(K-1)*NDR
JKP=JK+NDR

```

39760
39770
39780
39790
39800
39810
39820
39830
39840
39850
39860
39870
39880
39890
39900
39910
39920
39930
39940
39950
39960
39970
39980
39990
40000
40010
40020
40030
40040
40050
40060
40070
40080
40090
40100
40110
40120
40130
40140
40150
40160
40170
40180
40190
40200
40210
40220
40230


```

70 LV=4
   LLV=NDR*NDZ*(IV(LLV)-1)
   DXMP=DXM/(DXP*(DXM+DXP))
   DXPM=DXP/(DXM*(DXM+DXP))
   DXPM=(DXP-DXM)/(DXP*DXM)
   IVLV=IV(LLV)
   DP=2.*COA*DX2*DT*DXMP*ABS (VAR(JKP+LLV)-VAR(JK+LLV))/(DXP*DXP)
   DM=2.*COA*DX2*DT*DXPM*ABS (VAR(JK+LLV)-OV(IVLV))/(DXM*DXM)
   D=0.
   IF(ABS (1.-DXM/DXP).LT.10.E-6) GO TO 85
   D=2.*COA*DX2*DT*DXPM*ABS (DXMP*VAR(JKP+LLV)+DXPM*VAR(JK+LLV)-DXP
1M*OV(IVLV))
85 CONTINUE
   DO 90 JJ=1,NV
   LLV=NDR*NDZ*(JJ-1)
   OVM(JJ)=OV(JJ)
   VAR(JK+LLV)=VAR(JK+LLV)+DP*(VAR(JKP+LLV)-OV(JJ))-OVM(JJ)
   IF(D.EQ.0.) GO TO 90
   VAR(JK+LLV)=VAR(JK+LLV)+D*(DXMP*VAR(JKP+LLV)+DXPM*OV(JJ)-DXPM*OVM
1(JJ))
90 CONTINUE
100 RETURN
END

```

```

SUBROUTINE SPECHK(J,K)
THIS SUBROUTINE USES FILES DATA, RECORD, AND OLDREC
INPUT CARD NUMBERS 4, 5, AND 6.
COMMON/C1/DUM(8),ITCVER,CLNLM2,DUMDUM(8)
COMMON/C3/DAT(175),DADT(25),VAR(3500)
COMMON/C6/ISTEP,IV(10),IVV(10),IAV(10)
COMMON/C8/TCZM(4),TCRM(4),TC(4),TCRP(4),TCZP(4),DTCDR(4),DTCDZ(4),
1 DTCDVM(4,10),DTCDV(4,10),DTCDVP(4,10),DTDVDX(4,10),W(4),
2 W(4)
COMMON/C9/VARIZM(10),VARIRM(10),VARIRP(10),VARIZP(10),
1 DVDR(10),DVDZ(10),D2VDP2(10),D2VDZ2(10)
COMMON/C10/RM,R,RP,DRM,DRP,DRMP,DRPM,DDRPM,
1 ZM,Z,ZP,DZM,DZP,DZMP,DZPM,DDZPM
COMMON/C11/BEZM(4),BERM(4),BE(4),BEZP(4),BEPZM(4),BEPRM(4)
1 BEP(4),BEPRP(4),BEPZP(4),DBEDR(4),DBEDZ(4),DBEPDR(4),
2 DBEPDZ(4)
COMMON/C13/IDDX(10)
COMMON/C15/ICDZM(4),ICDRM(4),ICD(4),TCDRP(4),TCDZP(4)
COMMON/C20/INDA,NRZ,NIV,NAU,NAF,NAE
COMMON/C23/IFLAG,INCOM23
COMMON/C25/CVRMZP(10),CVRMZM(10),CVRPZP(10),CVRPZM(10)

```

C C


```

COMMON/C26/  BERMZP(04), BERPZP(04), BERMZM(04), BERPZM(04)
COMMON/C27/  TCRMZP(04), TCRPZP(04), TCRMZM(04), TCRPZM(04),
1  TDRMZP(04), TDRPZP(04), TDRMZM(04), TDRPZM(04)
COMMON/C30/  INP, IOUT1, IOUT2, INP2, IOUT4
DIMENSION  CVAR(10,4), CBE(4,4), CTC(4,4), CTD(4,4), COM23(5),
1  TTCD(4,5), U(10), VVARI(10,5), DBE(4,4),
2  TTC(4,5),
EQUIVALENCE (COM23(1), IFLAG),
EQUIVALENCE (CVAR, CVMZP), (CBE, BERMZP), (CTD, TDRMZP), (CTC, TCRMZP)
EQUIVALENCE (DADT(1), TIME), (DADT(2), NNDT),
EQUIVALENCE (DA(1), NDIM), (DA(4), NTC), (DA(5), IDATE), (DA(9), NDR),
1  (DA(10), NDZ), (DA(11), DT), (DA(16), NV), (DA(19), IFYL),
EQUIVALENCE (DA(23), ITAPE), (DA(24), IBE), (DA(55), IDVDX),
2  (DA(165), ISO),
3  (VVARI(1), VARIZM(1)), (TTC(1), TCZM(1)),
EQUIVALENCE (TTCD(1), TCZM(1)), (EB(1), BEZM(1)), (DV(1), DVDR(1)),
1  (DBE(1), DBEDR(1)), (DTCDM(1), DDCDV(1))
EQUIVALENCE (VAR, U)
JP=J+1
JM=J-1
KP=K+1
KM=K-1
10  FORMAT(' ', ' ')
20  WRITE( IOUT2, 25) ITAPE, IEYL, IDATE, TIME, NNDT, DT
25  FORMAT(1, F10.4, 14, '-', 12, 5X, A8, 5X, 'TIME=', E13.6, 5X, 'TIME STEP=', 15
1, 5X, DT =, E13.6)
IF(NDIM.EQ.1) WRITE( IOUT2, 30) JM, K, J, K, JP, K
30  IF(NDIM.EQ.2) WRITE( IOUT2, 30) J KM, JM, K, J, K, JP, K, J, KP
1, 12, ' ')
MM=3-NDIM
MMM=3+NDIM
DO 50 N=MM, MMM
DO 40 L=1, NV
40  OUTPUT(L)=VVARI( IVV(L), M)
50  WRITE( IOUT2, 20) (OUTPUT(L), L=1, NV)
IF(ISO.NE.1) GO TO 54
IF(IV(10).GT.NV) GO TO 54
WRITE( IOUT2, 10)
DO 53 M=MM, MMM
DO 50 N=MM, MMM
OUTPUT(1)=VVARI(7, M)
OUTPUT(2)=VVARI(9, M)
53  WRITE( IOUT2, 20) OUTPUT(1), OUTPUT(2)
54  CONTINUE
WRITE( IOUT2, 10)
DO 60 N=MM, MMM
60  WRITE( IOUT2, 20) (TTC(L, M), L=1, NTC)

```

40680
40690
40700
40710
40720
40730
40740
40750
40760
40770
40780
40790
40800
40810
40820
40830
40840
40850
40860
40870
40880
40890
40900
40910
40920
40930
40940
40950
40960
40970
40980
40990
41000
41010
41020
41030
41040
41050
41060
41070
41080
41090
41100
41110
41120
41130
41140
41150


```

41160 IF(ISO.NE.1) GO TO 65
41170 WRITE(IOUT2,10)
41180 DO 62 M=MM,MMM
41190 WRITE(IOUT2,20) (TTCD(L,M),L=1,NTC)
41200 CONTINUE
41210 IF(INDIM.EQ.1) GO TO 535
41220 IF(ISO.NE.1) GO TO 535
41230 WRITE(IOUT2,30) JM,KP,JP,KP,JM,KM,JP,KM
41240 DO 510 M=1,4
41250 L=1,10
41260 OUTPUT(L)=CVAR(L,M)
41270 WRITE(IOUT2,20) (OUTPUT(L),L=1,10)
41280 CONTINUE
41290 DO 520 M=1,4
41300 L=1,NTC
41310 OUTPUT(L)=CTC(L,M)
41320 WRITE(IOUT2,20) (OUTPUT(L),L=1,NTC)
41330 CONTINUE
41340 DO 530 M=1,4
41350 L=1,NTC
41360 OUTPUT(L)=CTD(L,M)
41370 WRITE(IOUT2,20) (OUTPUT(L),L=1,NTC)
41380 CONTINUE
41390 IF(IBE.NE.1) GO TO 75
41400 WRITE(IOUT2,72)
41410 FORMAT(/,5X,'EXTERNAL MAGNETIC FIELD COMPONENTS')
41420 DO 74 M=MM,MMM
41430 LLL=0
41440 DO 73 L=1,4
41450 GO TO (67,68,67,68),L
41460 IF(IV(10).LE.NV).AND.(ISO.EQ.1) GO TO 69
41470 IF(IV(L+6).GT.NV) GO TO 73
41480 CONTINUE
41490 DO 71 LL=1,2
41500 OUTPUT(LLL+LL)=EB(L,M+5*(LL-1))
41510 LLL=LLL+2
41520 CONTINUE
41530 WRITE(IOUT2,20) (OUTPUT(L),L=1,LLL)
41540 IF(ISO.NE.1) GO TO 75
41550 IF(INDIM.EQ.1) GO TO 75
41560 WRITE(IOUT2,10)
41570 DO 550 M=1,4
41580 L=1,4
41590 OUTPUT(L)=CBE(L,M)
41600 WRITE(IOUT2,20) (OUTPUT(L),L=1,4)
41610 CONTINUE
41620 IF(ISO.NE.1) GO TO 84
41630 IF(INDIM.EQ.1) GO TO 84

```



```

76 GO IQ(76,77,76,77), ISTEP
   WRITE(IOUT2 ,78) JM,KM,J,KM,JP,KM
   JK=J
77 GO TO 79
   WRITE(IOUT2 ,78) JM,KM,JM,K,KM,KP
78 FORMAT(/,5X,'NEW VARIABLES AT ',3(' ',12,' ',12,' '),5X))
79 WRITE(IOUT2 ,10)
   DO 83 L=1,3
   LL=JK-2+L
   DO 82 M=1,NV
82 OUTPUT(M)=U(NAU+M+NV*(LL-1))
83 WRITE(IOUT2 ,20) (OUTPUT(M),M=1,NV)
84 CONTINUE
   WRITE(IOUT2 ,80) (IDDX(L),L=1,10)
80 FORMAT(/,5X,'VARIABLE AND TRANSPORT COEFFICIENT SPATIAL DERIVATIVE
   1S, 10X,10I11)
   IF(IDDX.EQ.1) WRITE(IOUT2 ,81)
81 FORMAT(5X,'VARIABLE DERIVATIVES ARE FOURTH ORDER')
   DO 90 L=1,NDIM
   IF(L.EQ.2) WRITE(IOUT2 ,10)
   DO 90 M=1,2
   DO 85 N=1,NV
   NN=IV(N),NV
85 OUTPUT(N)=DV(NN ,20) ,L+2*(M-1))
90 CONTINUE
   WRITE(IOUT2 ,10)
   WRITE(IOUT2 ,20) (DTCDR(L),L=1,NTC)
   IF(NDIM.EQ.2) WRITE(IOUT2 ,20) (DTCNZ(L),L=1,NTC)
   IF(IRE.NE.1) GO TO 105
   WRITE(IOUT2 ,102)
102 FORMAT(/,5X,'EXTERNAL MAGNETIC FIELD DERIVATIVES')
   WRITE(IOUT2 ,10)
   DO 100 L=1,NDIM
   IF(L.EQ.2) WRITE(IOUT2 ,10)
   LL=0
   DO 98 M=1,4
   IF(IV(M+6).GT.NV) GO TO 98
   DO 97 N=1,2
97 OUTPUT(LL+N)=DBE(M,L+2*(N-1))
   LL=LL+2
98 CONTINUE
100 WRITE(IOUT2 ,20) (OUTPUT(N),N=1,LL)
105 CONTINUE
110 FORMAT(/,5X,'TRANSPORT COEFFICIENT DERIVATIVES, QUANTITIES')
   WRITE(IOUT2 ,111) ITCVER,CLNLM2

```



```

111 FORMAT(1X,11,15X,E17.9)
    WRITE(IOUT2,10)
    DO 115 N=1,4
    DO 115 M=1,NTC
    DO 115 L=1,NV
    LL=IVV(L)
    IF(DDTCDV(M,LL,N).NE.0.) GO TO 120
    CONTINUE
115 WRITE(IOUT2,116)
116 FORMAT(10X,'DERIVATIVES ARE ZERO')
    WRITE(IOUT2,10)
    GO TO 150
120 DO 140 N=1,4
    DO 130 M=1,NTC
    DO 125 L=1,NV
    LL=IVV(L)
    OUTPUT(LL)=DDTCDV(M,LL,N)
125 WRITE(IOUT2,20) (OUTPUT(L),L=1,NV)
130 WRITE(IOUT2,10)
140 WRITE(IOUT2,10)
    CONTINUE
    WRITE(IOUT2,20) (W(L),L=1,NTC)
    WRITE(IOUT2,20) (WW(L),L=1,NTC)
    IF(FLAG.NE.1) GO TO 165
    WRITE(IOUT2,160)
160 FORMAT(/,5X,'COMMON/23/ QUANTITIES')
    WRITE(IOUT2,20) (COM23(L),L=3,NCOM23)
165 CONTINUE
    WRITE(IOUT2,200)
200 FORMAT(/,5X,'MESH QUANTITIES')
    WRITE(IOUT2,20) RM,RP,DRM,DRP,DRMP,DRPM,DDRPM
    IF(NDIM.EQ.2) WRITE(IOUT2,20) ZM,Z,RP,DZM,DZP,DZMP,DZPM,DDZPM
    RETURN
END

SUBROUTINE STIRUP(ISUB,IPU,JJ,KK)
THIS SUBROUTINE USES FILES RECORD,(10), DATA (20) AND OLDREC (30)
COMMON/C3/ DA(175),DADT(25),VAR(3500)
COMMON/C4/ IBCRR(10),IBCR(10),IBCZ(10),IBSZ(10),ISHK(10),NZP(3),
1 IMPSHK,JSTEP
COMMON/C6/ ISTEP,IV(10),IVV(10),IAV(10)
COMMON/C20/ NDA,NRZ,NTV,NAT,NAU,NAF,NAE
COMMON/C21/ RZ(150)
COMMON/C22/ JDDZ(10),JDDZ(10)
COMMON/C30/ INP,IOUT1,IOUT2,INP2,IOUT4
DIMENSION JBCRR(2),JBCR(2),JBCZ(2),JBCZ(2),ISHOCK(2)
EQUIVALENCE {DA(1),NDIM},{DA(2),IVERS},{DA(3),MAT},{DA(4),ITC},
1 {DA(5),IDATE},{DA(6),JBCR},{DA(8),NDT},

```

42120
42130
42140
42150
42160
42170
42180
42190
42200
42210
42220
42230
42240
42250
42260
42270
42280
42290
42300
42310
42320
42330
42340
42350
42360
42370
42380
42390
42400
42410
42420
42430
42440

42450
42460
42470
42480
42490
42500
42510
42520
42530
42540
42550
42560

C


```

2      (DA(9),NDR),(DA(10),NDZ),(DA(11),DT),(DA(12),DRMIN),
3      (DA(13),DZMIN),(DA(14),IPVOT),(DA(15),ISTEPO),
4      (DA(16),NV),(DA(17),IFROM),(DA(18),STRIM),
5      (DA(19),IFYL),(DA(20),NADA),(DA(21),IV),
6      (DA(22),IIV),(DA(23),ITAPE),(DA(24),IBE),
7      (DA(25),RMIN),(DA(26),CQA),(DA(27),IAV),
8      (DA(28),IPLAVC),(DA(29),INTCON),(DA(30),JBCZZ),
9      (DA(31),NDIMO),(DA(32),JDDR),(DA(33),LASTDT),
A      (DA(34),JBCRR),(DA(35),JBCZ),(DA(36),JDDZ),
B      (DA(37),ISHOCK)
C      (DA(38),TIME),(DA(39),NNDT),(DA(40),ISTPNO),
D      (DA(41),INEG),(DA(42),MMAT),(DA(43),JSTPNO)
E      (DA(44),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
F      JBCRR,JBCZ,JDDZ,IST)
G      (DA(45),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
H      JBCRR,JBCZ,JDDZ,IST)
I      (DA(46),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
J      JBCRR,JBCZ,JDDZ,IST)
K      (DA(47),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
L      JBCRR,JBCZ,JDDZ,IST)
M      (DA(48),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
N      JBCRR,JBCZ,JDDZ,IST)
O      (DA(49),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
P      JBCRR,JBCZ,JDDZ,IST)
Q      (DA(50),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
R      JBCRR,JBCZ,JDDZ,IST)
S      (DA(51),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
T      JBCRR,JBCZ,JDDZ,IST)
U      (DA(52),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
V      JBCRR,JBCZ,JDDZ,IST)
W      (DA(53),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
X      JBCRR,JBCZ,JDDZ,IST)
Y      (DA(54),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
Z      JBCRR,JBCZ,JDDZ,IST)
AA     (DA(55),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
AB     JBCRR,JBCZ,JDDZ,IST)
AC     (DA(56),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
AD     JBCRR,JBCZ,JDDZ,IST)
AE     (DA(57),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
AF     JBCRR,JBCZ,JDDZ,IST)
AG     (DA(58),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
AH     JBCRR,JBCZ,JDDZ,IST)
AI     (DA(59),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
AJ     JBCRR,JBCZ,JDDZ,IST)
AK     (DA(60),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
AL     JBCRR,JBCZ,JDDZ,IST)
AM     (DA(61),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
AN     JBCRR,JBCZ,JDDZ,IST)
AO     (DA(62),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
AP     JBCRR,JBCZ,JDDZ,IST)
AQ     (DA(63),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
AR     JBCRR,JBCZ,JDDZ,IST)
AS     (DA(64),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
AT     JBCRR,JBCZ,JDDZ,IST)
AU     (DA(65),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
AV     JBCRR,JBCZ,JDDZ,IST)
AW     (DA(66),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
AX     JBCRR,JBCZ,JDDZ,IST)
AY     (DA(67),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
AZ     JBCRR,JBCZ,JDDZ,IST)
BA     (DA(68),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
BB     JBCRR,JBCZ,JDDZ,IST)
BC     (DA(69),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
BD     JBCRR,JBCZ,JDDZ,IST)
BE     (DA(70),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
BF     JBCRR,JBCZ,JDDZ,IST)
BG     (DA(71),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
BH     JBCRR,JBCZ,JDDZ,IST)
BI     (DA(72),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
BJ     JBCRR,JBCZ,JDDZ,IST)
BK     (DA(73),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
BL     JBCRR,JBCZ,JDDZ,IST)
BM     (DA(74),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
BN     JBCRR,JBCZ,JDDZ,IST)
BO     (DA(75),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
BP     JBCRR,JBCZ,JDDZ,IST)
BQ     (DA(76),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
BR     JBCRR,JBCZ,JDDZ,IST)
BS     (DA(77),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
BT     JBCRR,JBCZ,JDDZ,IST)
BU     (DA(78),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
BV     JBCRR,JBCZ,JDDZ,IST)
BW     (DA(79),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
BX     JBCRR,JBCZ,JDDZ,IST)
BY     (DA(80),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
BZ     JBCRR,JBCZ,JDDZ,IST)
CA     (DA(81),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
CB     JBCRR,JBCZ,JDDZ,IST)
CC     (DA(82),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
CD     JBCRR,JBCZ,JDDZ,IST)
CE     (DA(83),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
CF     JBCRR,JBCZ,JDDZ,IST)
CG     (DA(84),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
CH     JBCRR,JBCZ,JDDZ,IST)
CI     (DA(85),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
CJ     JBCRR,JBCZ,JDDZ,IST)
CK     (DA(86),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
CL     JBCRR,JBCZ,JDDZ,IST)
CM     (DA(87),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
CN     JBCRR,JBCZ,JDDZ,IST)
CO     (DA(88),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
CP     JBCRR,JBCZ,JDDZ,IST)
CQ     (DA(89),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
CR     JBCRR,JBCZ,JDDZ,IST)
CS     (DA(90),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
CT     JBCRR,JBCZ,JDDZ,IST)
CU     (DA(91),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
CV     JBCRR,JBCZ,JDDZ,IST)
CW     (DA(92),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
CX     JBCRR,JBCZ,JDDZ,IST)
CY     (DA(93),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
CZ     JBCRR,JBCZ,JDDZ,IST)
DA     (DA(94),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
DB     JBCRR,JBCZ,JDDZ,IST)
DC     (DA(95),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
DD     JBCRR,JBCZ,JDDZ,IST)
DE     (DA(96),IPLAVC,IPU,IVERS,IPVOT,ISTEPO,NDT,NDTPNT,NDR,NDZ,
DF     JBCRR,JBCZ,JDDZ,IST)

```



```

IF(IBE.EQ.1) CALL BEXINT(1)
JJ=2
KK=1 PNT
ISUB=NDTPNT
GO TO 120
78 CONTINUE
IFROMO=IFROM
IFYLO=IFYL
ITP=INP2
IIPMW=IMP2MW
REWIND ITP
ITAPEO=ITAPE
CALL BUFFER(0,1,ITP)
NIV=NV*NDR*NDZ
NRZ=NDR*NDZ
CALL BUFFER(0,2,ITP)
READ(ITP) DADT
CALL BUFFER(0,3,ITP)
IF(NNDT,LT,IST) GO TO 92
ITP=IOU*1100*ITAPE+IFROMO
IFROMO=100*ITAPE
ITAPE=ITAPEO
IFROM=IFROMO
ISTEP=ISTPNO
JSTEP=JSTEPNO
IDATE=IDATEO
STRTYM=TIME
IFYL=IFYLO
IF(1)
  READ(INP,107) IVERS,IPIVOT,NDT,NDTPNT,JBCRR,JBCZ,JBCZ,JBCZ,
  107 FORMAT(2X,2I1,1X,2I5,10X,10I5)
  IF(1 IVERS.EQ.0) IVERS=1
  GO TO 120
110 CONTINUE
IF(1 STEPONE=0) GO TO 130
111 GO TO (112,113,113,112),IVERS
112 ISTEP=2
  GO TO 140
113 ISTEP=4
  GO TO 140
130 GO TO (131,132,131,131),IVERS
131 ISTEP=1
  GO TO 140
132 ISTEP=3
  GO TO 140
140 CONTINUE
TIME=0
STRTYM=0

```



```

120 IFROM=0
CONTINUE
IMPSHK=0
DO 166 K=1,2
DO 166 J=1,5
K1=1
K1=11
IF(K.EQ.1) K1=6
ISHK(K1-J)=ISHOCK(K)/(10**(J-1))-10*(ISHOCK(K)/(10**J))
IBCR(K1-J)=JBCR(K)/(10**(J-1))-10*(JBCR(K)/(10**J))
JBCZ(K1-J)=JBCZ(K)/(10**(J-1))-10*(JBCZ(K)/(10**J))
JBCRR(K1-J)=JBCRR(K)/(10**(J-1))-10*(JBCRR(K)/(10**J))
JBCZZ(K1-J)=JBCZZ(K)/(10**(J-1))-10*(JBCZZ(K)/(10**J))
IF(ISHK(K1-J).GT.5) IMPSHK=ISHK(K1-J)
166 CONTINUE
IF(IPU.NE.0) RETURN
CALL MATR1X
ITCDAT=0
IF(1FROM.NE.0) ITCDAT=1
CALL TCINIT(ITCDAT)
IF(IBE.EQ.0) GO TO 164
NTV=NV*NDR*NDZ
IBEDAT=0
IF(1FROM.NE.0) IBEDAT=1
CALL BEXINT(IBEDAT)
CONTINUE
IF(1VERS.EQ.0) 1VERS=1
TIME=STR1YM-DT
IPNT=NDTPNT-1
INEG=0
NNDT=0
KK=1PNT
ISUB=NDTPNT
JJ=1
RETURN
JJ=3
RETURN
END

1060
SUBROUTINE TCINIT(ITCDAT)
THIS SUBROUTINE USES INPUT CARD NUMBER 9
ALL REAL#8 COMT1,COMT2,COMT3,COMT4,COMT5,COMT6,IDA
COMMON/CL/GFEQ1,GFEQ2,GRES1,CRES2,CKI1,CKI2,CKE1,CKE2,ITCVER,
1 CLNMW2,CKIPI1,CKE11,EMASS,PMASS,UO,GAM1,GAM2,PI,TCRO
COMMON/C3/ DA(17.5),DADT(25),VAR(35.00)
COMMON/C30/ INP,IOU1,XXX,IOU2,INP2,YYY,IOU4
DIMENSION IDA(40),ADDA(40)
43900
43910
43920
43930
43940
43950
43960
43970
43980
43990

```

```

C C
THIS SUBROUTINE ICINIT(I,TCDAT)
ALL QUANTITIES IN MKS UNITS CARD NUMBER 9
REAL*8 COM1,COM2,COM3,COM4,COM5,COMT6,IDA
REAL*8 COM7,CFEQ1,CFEQ2,CRESQ1,CRES2,CKI1,CKI2,CKE1,CKE2,ITCVER,
      COM8/C1/DLAL(17.5),DAD1(25),VAR(3500)
1 COMMON/C3/ CNLM2,C INP,IOUT1,XXX,IOUT2,INP2,YYY,IOUT4
COMMON/C30/ DIMENSION IDA(40),ADDA(40)
43900
43910
43920
43930
43940
43950
43960

```



```

EQUIVALENCE (DA(4),NTC),(DA(16),NV),(DA(20),NADDA),(DA(26),ADDA(1)),(DA(71),IDA(1)),(DA(158),NTCDRV)
EQUIVALENCE (DADT(6)),IITC)
DATA COM11,TC VERS=,/,COMT2/, GAM=,/,COMT3/, K,KP=,/,
COMT4/, PMASS=,/,COMT5/, CLNLM2=,/,COMT6/, TCRO=,/,
IITC=4
IITC=2
STKEQO=0.
GAM=0.
PMASS=0.
IF(IITCDAT.EQ.1) GO TO 15
READ(INP,10) ITCVER,CLNLM2,TCRO,GAM,PMASS,STKEQO
FORMAT(4X,11,7E10.3)
IF( STKEQO.EQ.0.) GO TO 11
NADDA=NADDA+1
IDDA(NADDA)=COMT3
ADDA(NADDA)=0.
CONTINUE
11 NADDA=NADDA+1
IDDA(NADDA)=COMT1
ADDA(NADDA)=FLOAT(ITCVER)
NADDA=NADDA+1
IDDA(NADDA)=COMT5
ADDA(NADDA)=CLNLM2
IF(IITCVER.LT.4) GO TO 13
NADDA=NADDA+1
IDDA(NADDA)=COMT6
ADDA(NADDA)=TCRO
13 IF(GAM.EQ.0.) GO TO 14
NADDA=NADDA+1
IDDA(NADDA)=COMT2
ADDA(NADDA)=GAM
14 IF(PMASS.EQ.0.) GO TO 25
NADDA=NADDA+1
IDDA(NADDA)=COMT4
ADDA(NADDA)=PMASS
GO TO 25
15 CONTINUE
DO 20 J=1,NADDA
IF(IDA(J).NE.COMT1) GO TO 16
ITCVER=FIX(ADDA(J))
CLNLM2=FIX(ADDA(J+1))
IF(ITCVER.GE.4) TCRO=ADDA(J+2)
16 IF(IDA(J).EQ.COMT2) GAM=ADDA(J)
IF(IDA(J).EQ.COMT4) PMASS=ADDA(J)
IF(IDA(J).EQ.COMT3) STKEQO=1.
20 CONTINUE
25 CONTINUE

```

43970
43980
43990
44000
44010
44020
44030
44040
44050
44060
44070
44080
44090
44100
44110
44120
44130
44140
44150
44160
44170
44180
44190
44200
44210
44220
44230
44240
44250
44260
44270
44280
44290
44300
44310
44320
44330
44340
44350
44360
44370
44380
44390
44400
44410
44420
44430
44440


```

IF((ITCVER.EQ.3).OR.(ITCVER.EQ.5)) NTCDRV=1
PI=3.1415927
UO=4.*PI*1.E-07
EO=8.8541*1.E-12
IF(GAM.EQ.0.) GAM=5./3.
GAM1=GAM-1.
GAM2=GAM-2.
EMASS=9.1083*1.E-31
IF(PMASS.NE.0.) GO TO 30
PMASS=1.67239*2.E-27
30 CONTINUE
SMASS=EMASS+PMASS
CFEQ1=1.60206*1.E-19
CFEQ1=ECHG*ECHG*ECHG/(12.*PI*EO*SMASS*SMASS)*ECHG*SQRT(2.*
1/EMASS/(PI*SMASS)*PMASS)
C SEE CFEQ2=144.*PI*PI*EO*EO*SMASS/(ECHG**3)/(ECHG**3)*SMASS*SMASS*
1/SMASS
C SEE EQUATION 2.3-7C
CRES1=ECHG*ECHG*SQRT(EMASS/(2.*PI*SMASS)))/(128.*EO*EO*SMASS)
IF(ITCVER.EQ.0.OR.ITCVER.EQ.6) CRES1=CRES1/0.582
C SEE EQUATION 2.3-13B
IF(ITCVER.EQ.1) CRES1=CRES1*32./3./PI
C SEE EQUATION 2.3-12B
CRES2=CFEQ2
CKE1=1300.*GAM1*EO*EO*SMASS/ECHG/ECHG/ECHG*SQRT(2.*PI*SMASS/EMASS)
1/ECHG*SMASS*SMASS
CKE1=0.225*0.419*CKE1
C SEE EQUATION 2.3-8B WHERE ALPHA=E
CKE2=CFEQ2
CKE1=1280.*GAM1*EO*EO*SMASS/ECHG/ECHG/ECHG*SQRT(2.*PI*SMASS/PMASS)
1/ECHG*SMASS*SMASS
CKE1=0.225*0.419*CKE1
C SEE EQUATION 2.3-8B WHERE ALPHA=I
CKE12=CFEQ2
CKE12=GAM1*ECHG*ECHG*SQRT(PMASS/PI/SMASS)/(12.*PI*EO*EO*SMASS)
C SEE EQUATION 2.3-9B
C SEE CKEPI=CKEPI*SQRT (EMASS/PMASS/2.)*(SQRT (2.)+13./4.)
C SEE EQUATION 2.3-9C
IF(SKEQ.EQ.0.) GO TO 50
CKE1=0.
CKE12=0.
CKEPI=0.
CKEPI=0.
50 CONTINUE
RETURN
END

```

44450
44460
44470
44480
44490
44500
44510
44520
44530
44540
44550
44560
44570
44580
44590

44600
44610

44620
44630

44640

44650
44660
44670
44680

44690
44700
44710
44720

44730
44740

44750

44760
44770
44780
44790
44800
44810
44820
44830


```

C
SUBROUTINE TEPL0T
THIS SUBROUTINE USES FILE MDHOUT.
COMMON/C3/,DA(175),DADT(25),VAR(3500)
COMMON/C6/,ISTEP,IV(10),IVV(10),IAV(10)
COMMON/C21/RZ(150)
COMMON/C30/,INP,IOUT1,IOUT2,INP2,IOUT4
DIMENSION A(600),R(600),NP(5),KZ(3),IPT(3)
EQUIVALENCE (DA(1),NDIM),(DA(5),IDATE),(DA(9),NDR),(DA(10),NDZ),
1 EQUIVALENCE (DADT(1),TIME),(DADT(2),NNDT)
DATA IPT/,*,*,*,*,*/
NZ=1
IF (NDZ.GT.50) NZ=2
NDRM=NDR-1
NP(1)=NDRM
NP(2)=NDRM
NP(3)=NDRM
NP(4)=0
NP(5)=0
WRITE(IOUT2,10) ITAPE,IFYL,IDATE,NNDT,TIME
FORMAT('ELECTRON TEMPERATURE',5X,15,'-',12,5X,A4,9X,15,5X,E13.6)
10 K=2-NZ
NG=(NDZ-1)/3
IF (NG.GT.8) NG=8
DO 120 L=1,NG
DO 100 M=1,3
K=K+NZ
KZ(M)=K
DO 100 J=2,NDR
JM=J-1+NDR*(M-1)
JKTE=J+NDR*(K-1+NDZ*(IV(6)-1))
R(JM)=RZ(J)
IF (IAV(6).EQ.1) GO TO 90
A(JM)=VAR(JKTE)
GO TO 100
90 JKRO=J+NDR*(K-1)
A(JM)=VAR(JKTE)/VAR(JKRO)
100 CONTINUE
WRITE(IOUT2,110) (IPT(M),KZ(M),M=1,3)
110 FORMAT(5X,3(A1,' AT Z=',12,5X))
CALL PLOT1R(A,R,NP,RMAX)
120 CONTINUE
RETURN
END

```

44840
44850
44860
44870
44880
44890
44900
44910
44920
44930
44940
44950
44960
44970
44980
44990
45000
45010
45020
45030
45040
45050
45060
45070
45080
45090
45100
45110
45120
45130
45140
45150
45160
45170
45180
45190
45200
45210
45220
45230
45240
45250


```

200 FEQ=CR*ES1*LNL2E/(TE*RT2TE)
400 TE(ISO,NE,1) GO TO 4010
BR=VARI(7)
BP=VARI(8)
BZ=VARI(9)
BPE=BE(1)
BZE=BE(2)
BZL=BE(3)
BZL=BR*BR+BP*BZ+BR*BR*BR+BPE*BZ*BR*BR*BR
KIPINV=R2*RT2TE/(CKEPI*RO2*LNL2E)
KIP=K1/(1.+K1*KIPINV)
KIP=KE/(1.+KE*KEPINV)
KID=KI-KIP
KED=KE-KEP
KI=KIP
KE=KEP
RESD=-RES*0.98
RES=1.98*RES
TCD(1)=KID
TCD(2)=KED
TCD(4)=RESD
4010 CONTINUE
IF(NTCDRV.EQ.1) GO TO 5000
IF(NTCDRV.EQ.1) GO TO 5000
IF(NTCDRV.EQ.3) OR(NTCDRV.EQ.5) GO TO 5000
Q1=LAM2I/LNL2E/(CKEPI*RO2*LNL2E)
QE=LAM2E/LNL2E/(CLNL2+LAM2E)
DTCDV(1,1)=K1*Q1/RO
DTCDV(1,5)=K1*(2.5-3.*Q1)/TI
DTCDV(2,1)=KE*QE/RO
DTCDV(2,6)=KE*(2.5-3.*QE)/TE
DTCDV(3,1)=FEQ*(1.-Q1)/RO
DTCDV(3,5)=-1.5*FEQ*EMASS/(EMASS*TI+PMASS*TE)
DTCDV(3,6)=FEQ*(-1.5*PMASS/(EMASS*TI+PMASS*TE)+3.*QE/TE)
DTCDV(4,1)=-RES*Q1/RO
DTCDV(4,6)=RES*(-1.5*3.*QE)/TE
IF(IAV(1).NE.1) GO TO 4502
DO 4501 L=1,NTC
DTCDV(L,1)=-RO2*DTCDV(L,1)+RO*(TI*DTCDV(L,5)+TE*DTCDV(L,6))
4501 DTCDV(L,1)=DTCDV(L,1)
4502 IF(IAV(5).NE.1) GO TO 4504
DO 4503 L=1,NTC
DTCDV(L,5)=DTCDV(L,5)/RO
4503 DTCDV(L,5)=DTCDV(L,5)/RO
4504 IF(IAV(6).NE.1) GO TO 5000
DO 4505 L=1,NTC
DTCDV(L,6)=DTCDV(L,6)/RO
4505 DTCDV(L,6)=DTCDV(L,6)/RO
5000 CONTINUE

```


6000

CONTINUE
TC(1)=KI
TC(2)=KE
TC(3)=FEQ
TC(4)=RES
RETURN
END

46230
46240
46250
46260
46270
46280
46290

SUBROUTINE TRIANG(IF1)

INTEGER COL
COMMON/C3/DA(175),DADT(25),VAR(3500)
COMMON/C7/A(10,10),B(10,10),D(10,10),V(10),COL(10)
EQUIVALENCE (DA(14),IPIVOT),(DA(16),NV)
DO 20 J=1,NV
COL(J)=J

46300
46310
46320
46330
46340
46350
46360
46370
46380
46390
46400
46410
46420
46430
46440

IND=1
30 CONTINUE
IND1=IND+1
DO 110 L=IND1,NV
DO 100 M=IND1,NV
A(L,M)=A(L,M)-A(L,IND)*A(IND,M)/A(IND,IND)
IF (IF1.EQ.1) GO TO 107

46450
46460
46470
46480
46490
46500
46510
46520
46530
46540
46550
46560
46570
46580
46590
46600
46610
46620
46630
46640
46650
46660
46670
46680

DO 103 M=1,NV
B(L,M)=B(L,M)-A(L,IND)*B(IND,M)/A(IND,IND)
30 CONTINUE
V(L)=V(L)-A(L,IND)*V(IND)/A(IND,IND)
A(L,IND)=0
IF (IND1.GE.NV) GO TO 140
IND=IND1
GO TO 30

140 CONTINUE
V(NV)=V(NV)/A(NV,NV)
IF (IF1.EQ.1) GO TO 155
DO 150 L=1,NV
B(NV,L)=B(NV,L)/A(NV,NV)
150 NVN=NV-1
DO 190 L=1,NVN
MM=NV+1-L
N=NV-L

46690
46700
46710
46720
46730
46740
46750
46760
46770
46780
46790
46800
46810
46820
46830
46840
46850
46860
46870
46880
46890
46900
46910
46920
46930
46940
46950
46960
46970
46980
46990
47000

DO 170 M=MM,NV
V(N)=V(N)-A(N,M)*V(M)
IF (IF1.EQ.1) GO TO 170
DO 160 LL=1,NV
B(N,LL)=B(N,LL)-A(N,M)*B(M,LL)
160 A(N,M)=0
170 V(N)=V(N)/A(N,N)
IF (IF1.EQ.1) GO TO 185

46690
46700
46710
46720
46730
46740

DO 180 M=1,NV
B(N,M)=B(N,M)/A(N,N)
180 A(N,N)=1.
185 CONTINUE
190 RETURN
END

46750

SUBROUTINE ZBC
THIS SUBROUTINE USES FILE MHDOUT.
IMPLICIT REAL*8 (C)
REAL*8 P
COMMON/C4/ IBCR(10), IBCZ(10), ISHK(10), NZP(3),
COMMON/C6/ ISTEP, IV(10), IVV(10), IAV(10)
1 COMMON/C21/RZ(150)
COMMON/C30/ INP, IOUT1, IOUT2, INP2, IOUT4
DIMENSION P(50)
EQUIVALENCE (DA(11),NDIM), (DA(9),NDR), (DA(13),DZ),
(DA(16),NV), (DA(25),RMIN)
1 DATA COMT1//DRO/DZ=0//,COMT2//RO=MIN+//,COMT3//RO-MIN//,//,
2 COMT4//EXP(-DT//,COMT5//IRO//,COMT6//EXP(VZ*//,//,
3 COMT7//DT/XRO*//,COMT8//DZ//,//,COMT9//DVR/DZ=0//,//,
4 COMT10//VR=5*VR//,COMT11//(-//,COMT12//VR=0//,//,
5 COMT13//DVP/DZ=0//,COMT14//VP=0//,COMT15//VZ=0//,//,
6 COMT16//VZ=5*V//,COMT17//(-//,COMT18//DT/DZ=0//,//,
7 COMT19//DT/DZ=0//,COMT20//DTE/DZ=0//,COMT21//DTE/DZ=0//,//,
8 COMT22//BR=0//,COMT23//DBP/DZ=0//,COMT24//DBZ/DZ=0//,//,
COMT25//DSI/DZ=0//,COMT26//,COMT27//DVZ/DZ=0//,
DO 2030 K=1,2

C

46760
46770
46780
46790
46800
46810
46820
46830
46840
46850
46860
46870
46880
46890
46900
46910
46920
46930
46940
46950
46960
46970
46980
46990
47000
47010
47020
47030
47040
47050
47060
47070
47080
47090
47100
47110
47120
47130

DO 2010 J=1,NV
M=0
JJ=IVV(J)
IBZJ=IBZ(JJ)
IF (K=1) GO TO (100,200,300,400,500,600,700,800,900,1000),JJ
GO TO (150,250,350,450,550,650,750,850,950,1050),JJ
CONTINUE
100 P(M+1)=COMT1
101 M=M+1
GO TO 2000
150 GO TO (101,152,153),IBZJ
152 P(M+1)=COMT2
P(M+2)=COMT3
M=M+2
JZP=IBZJ+1
GO TO (160,153),JZP

160	P(M+1)=COMT4	47140
	P(M+2)=COMT5	47150
	M=M+2	47160
	GO TO 2000	47170
153	P(M+1)=COMT6	47180
	P(M+2)=COMT7	47190
	P(M+3)=COMT8	47200
	M=M+3	47210
	GO TO 2000	47220
200	CONTINUE	47230
201	P(M+1)=CGMT9	47240
	M=M+1	47250
	GO TO 2000	47260
250	GO TO (201,252,253), IBCZJ	47270
252	P(M+1)=COMT10	47280
	P(M+2)=COMT11	47290
	M=M+2	47300
	GO TO 2000	47310
253	P(M+1)=COMT12	47320
	M=M+1	47330
	GO TO 2000	47340
300	CONTINUE	47350
301	P(M+1)=COMT13	47360
	M=M+1	47370
	GO TO 2000	47380
350	GO TO (301,352,352), IBCZJ	47390
352	P(M+1)=COMT14	47400
	M=M+1	47410
	GO TO 2000	47420
400	CONTINUE	47430
401	P(M+1)=COMT15	47440
	M=M+1	47450
	GO TO 2000	47460
450	GO TO (401,452,453), IBCZJ	47470
452	P(M+1)=COMT17	47480
	M=M+1	47490
	GO TO 2000	47500
453	P(M+1)=COMT16	47510
	P(M+2)=COMT17	47520
	M=M+2	47530
	GO TO 2000	47540
500	CONTINUE	47550
501	P(M+1)=COMT18	47560
	M=M+1	47570
	GO TO 2000	47580
550	GO TO (501,552), IBCZJ	47590
552	P(M+1)=COMT19	47600
	M=M+1	47610


```

600 GO TO 2000
601 CONTINUE
    P(M+1)=COMT20
    M=M+1
650 GO TO 2000
652 GO TO (601,652),IBCZJ
    P(M+1)=COMT21
    M=M+1
700 GO TO 2000
701 CONTINUE
    P(M+1)=COMT22
    M=M+1
750 GO TO 2000
751 CONTINUE
800 GO TO 701
801 CONTINUE
    P(M+1)=COMT23
    M=M+1
850 GO TO 2000
851 CONTINUE
900 GO TO 801
901 CONTINUE
    P(M+1)=COMT24
    M=M+1
950 GO TO 2000
951 CONTINUE
1000 GO TO 901
1001 CONTINUE
    P(M+1)=COMT25
    M=M+1
1050 GO TO 2000
1051 CONTINUE
2000 GO TO 1001
2001 P(M+1)=COMT26
    M=M+1
2010 CONTINUE
    ZMIN=0
2020 IF (K.EQ.1) WRITE(IOUT,2,2020) ZMIN,(P(J),J=1,M)
    FORMAT(5X,'BOUNDARY CONDITIONS AT Z=,E12.5',--',3X,10A8/,3(7X,15
    1A8,/))
    ZMAX=(RZ(NDR+NDZ)+RZ(NDR+NDZ-1))/2.
2030 IF (K.EQ.2) WRITE(IOUT2,2020) ZMAX,(P(J),J=1,M)
    CONTINUE
    RETURN
    END

```

47620
47630
47640
47650
47660
47670
47680
47690
47700
47710
47720
47730
47740
47750
47760
47770
47780
47790
47800
47810
47820
47830
47840
47850
47860
47870
47880
47890
47900
47910
47920
47930
47940
47950
47960
47970
47980
47990
48000
48010
48020
48030
48040
48050
48060

LIST OF REFERENCES

1. Lindemuth, I. R., The Alternating-Direction Implicit Numerical Solution of Time-Dependent, Two-Dimensional, Two-Fluid Magnetohydrodynamic Equations, Ph.D. Thesis, University of California/Livermore, 1971.
2. Users Manual, 1st ed., W. R. Church Computer Center, 1970.
3. Control Data 6400/6500/6600 Computer Systems FORTRAN Extended Reference Manual, Revision G, Control Data Corporation, 1971.
4. 3400/3600/3800 Computer Systems FORTRAN Reference Manual, 1st ed., Control Data Corporation, 1965.
5. IBM System/360 FORTRAN IV Language, Form C28-6515-7, 8th ed., International Business Machines Corporation, 1968.
6. G. P. Learmonth, Naval Postgraduate School Computer Center, private communication (1972).
7. L. M. Brunner, Naval Postgraduate School Computer Center, private communication (1972).
8. User Libraries Under OS, Technical Note No. 0211-05, W. R. Church Computer Center, 1969.
9. Miscellaneous Computer Printouts concerning computer code described in UCRL 51103, by C. S. Yager, 1972.
10. IBM System/360 Operating System, Messages and Codes, C28-6631-7, 8th ed., 1969.

INITIAL DISTRIBUTION LIST

	No. Copies
1. Defense Documentation Center Cameron Station Alexandria, Virginia 22314	2
2. Library, Code 0212 Naval Postgraduate School Monterey, California 93940	2
3. Asst Professor G. A. Garrettson Code 61Gr Department of Physics Naval Postgraduate School Monterey, California 93940	1
4. Assoc Professor A. W. Cooper Code 61Cr Department of Physics Naval Postgraduate School Monterey, California 93940	1
5. Assoc Professor F. R. Schwirzke Code 61Sw Department of Physics Naval Postgraduate School Monterey, California 93940	1
6. Doctor I. R. Lindemuth 1277 Marguerite Street Livermore, California 94550	1
7. CPT Charles S Yager Jr 628 East Raymond Street Indianapolis, Indiana 46203	1

DOCUMENT CONTROL DATA - R & D

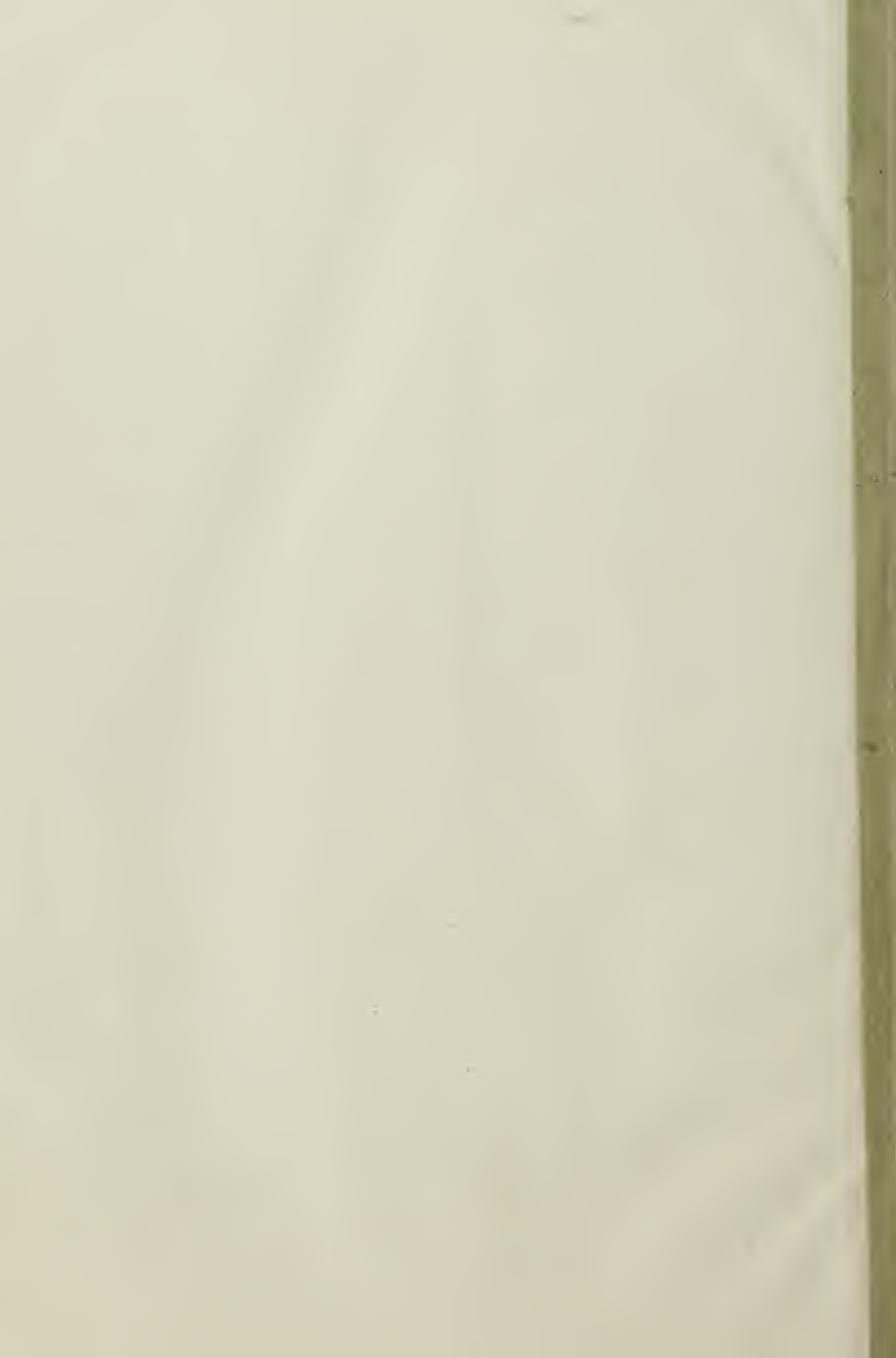
(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author) Naval Postgraduate School Monterey, California 93940		2a. REPORT SECURITY CLASSIFICATION Unclassified	
		2b. GROUP	
3. REPORT TITLE Conversion of a Two-Dimensional Magnetohydrodynamic Computer Code From CDC-6400 FORTRAN to IBM 360/67 FORTRAN			
4. DESCRIPTIVE NOTES (Type of report and, inclusive dates) Master's Thesis; June 1972			
5. AUTHOR(S) (First name, middle initial, last name) Charles S Yager Jr			
6. REPORT DATE June 1972		7a. TOTAL NO. OF PAGES 214	7b. NO. OF REFS 10
8a. CONTRACT OR GRANT NO.		9a. ORIGINATOR'S REPORT NUMBER(S)	
b. PROJECT NO.			
c.		9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)	
d.			
10. DISTRIBUTION STATEMENT Approved for public release; distribution unlimited.			
11. SUPPLEMENTARY NOTES		12. SPONSORING MILITARY ACTIVITY Naval Postgraduate School Monterey, California 93940	
13. ABSTRACT In his Ph.D. thesis Doctor Irvin R. Lindemuth, at the University of California/Livermore, developed a very general method for numerically solving two-dimensional, two-fluid magnetohydrodynamic equations. A copy of his computer code was given to the Physics Department at the Naval Postgraduate School for conversion to the IBM 360/67 system presently in operation at the school. This paper is intended to be a users manual for this code. Numerous changes to the original code were required due to the inherent differences between the CDC and IBM machines. The conversion of this code as well as a complete understanding of its operation and logic was the goal in the preparation of this paper.			

Unclassified

Security Classification

KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Computer Code						
Conversion						
MHD Equations						
Theta Pinch Computer Code						
Laser Produced Plasma						
Computer Code						



134813

Thesis

Y215

Yager

c.1

Conversion of a two-
dimensional magneto-

134813

Thesis

Y215

Yager

c.1

Conversion of a two-
dimensional magneto-
hydrodynamic computer
code from CDC-6400
FORTRAN to IBM 360/67
FORTRAN.

thesY215

Conversion of a two-dimensional magneto-



3 2768 001 90491 5

DUDLEY KNOX LIBRARY